CHALMERS
UNIVERSITY OF TECHNOLOGY

Metrics and methods for complexity analysis

Identifying driving factors of structural complexity in modular vehicle platforms

Master's thesis in Complex Adaptive Systems

Erik Stenmark Tullberg & Mattias Kvartsén

DEPARTMENT OF MECHANICS AND MARITIME SCIENCES

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2026
www.chalmers.se

Master's thesis 2026

Metrics and methods for complexity analysis

Identifying driving factors of structural complexity in modular vehicle platforms

Erik Stenmark Tullberg & Mattias Kvartsén

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Mechanics and Maritime Sciences
Division of Vehicle Engineering and Autonomous Systems
Chalmers University of Technology
Gothenburg, Sweden 2026

Metrics and methods for complexity analysis
Identifying driving factors of structural complexity in modular vehicle platforms
Erik Stenmark Tullberg & Mattias Kvartsén

Supervisor: Doctor Maria Siiskonen, Platform and Architecture team,
Trucks Technology & Industrial Division within Volvo Group
Supervisor: Erik Magnehov, Platform and Architecture team,
Trucks Technology & Industrial Division within Volvo Group
Supervisor: Professor Ola Isaksson, Product development,
Chalmers University of Technology
Examiner: Associate Professor Peter Forsberg, Vehicle Engineering and Autonomous Systems,
Chalmers University of Technology

Master's Thesis 2026
Department of Mechanics and Maritime Sciences
Division of Vehicle Engineering and Autonomous Systems
Chalmers University of Technology
SE-412 96 Gothenburg
Sweden
Telephone +46 31 772 1000

Cover: Line-up of different Volvo Trucks showing some of the diversity in Volvo's product portfolio. Each truck is configured differently to suit the customer's specific needs.

Typeset in LaTeX
Gothenburg, Sweden 2026

Metrics and Methods for Complexity Analysis in Modular Vehicle Architecture
A Quantitative Approach to Comparing Complexity in Modular Truck Platforms
Erik Stenmark Tullberg & Mattias Kvartsén
Department of Mechanics and Maritime Sciences
Chalmers University of Technology

Abstract

This thesis explores how module and interface variation can be used to quantify architectural complexity in modular vehicle platforms at Volvo Group TTI. By combining structural complexity theory with entropy-based assessment of module variant complexity, a quantitative methodology for architectural complexity assessment was developed. The proposed metric evaluates architectural complexity based on module connectivity, module and interface variation, and interdependencies between modules. Shannon entropy was adapted to assess module variant complexity. In addition, a data-processing pipeline was developed to enable large-scale analysis of industrial platform data. The results show that the proposed metric responds consistently to structural changes in the platform architecture and captures important characteristics associated with architectural complexity such as connectivity and interdependency. The framework therefore enables systematic and data-driven comparisons of modular platforms. The proposed metric should be used in combination with other metrics and methods to fully capture architectural complexity and support balanced and robust design decisions. The metric is designed for hardware platforms. Future studies could improve measurement accuracy of module variant complexity by analysing the physical structure of the module variants in detail. Additionally, future work could develop a systematic method for selecting interface complexity values.

Keywords: structural complexity, modularity, product architecture, vehicle platforms, Shannon entropy, product development.

Acknowledgements

We wish to express our utmost gratitude to Maria Siiskonen, for her guidance during this thesis. Thanks to her, questions were never left unanswered, and she always ensured that we stayed on track. With both sharp insight and warm encouragement, she greatly contributed to the quality of this work. We could not have wished for a better supervisor, and without her support, this thesis would not be what it is today. We also wish to express our gratitude to Erik Magnehov for his industrial expertise and advice throughout this thesis.
We wish to extend our appreciation to our examiner Peter Forsberg, for his dedication and substantial commitment to this thesis. His continuous aid provided inspiration and much needed structure. Further thanks are given to our supervisor at Chalmers, Ola Isaksson, for his assistance in theoretical and academic matters.

Finally, as this marks the end of our academic journeys, our heartfelt thanks go to our friends and families who have stood by us throughout these years. They have meant the world to us.

Erik Stenmark Tullberg, Gothenburg, June 22, 2026
Mattias Kvartsén, Gothenburg, June 22, 2026

Thesis advisor: Maria Siiskonen, Trucks Technology & Industrial Division within Volvo Group
Thesis advisor: Erik Magnehov, Trucks Technology & Industrial Division within Volvo Group
Thesis advisor: Ola Isaksson, Product Development
Thesis examiner: Peter Forsberg, Mechanics and Maritime Sciences

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

BEV	Battery Electric Vehicle
CSV	Comma-Separated Values
DSM	Design Struture Matric
EDB	Engineering Data Base
GIC	Generic Interface Component
GUI	Graphical User Interface
ICE	Internal Combustion Engine
KML	KOLA Module List
PC	Product Class
PDM	Product Data Management
RQ	Research Questions
SC metric	Structural Complexity metric
SIC	Specific Interface Component
TTI	Trucks Technology & Industrial Division within Volvo Group

Nomenclature

Below is the nomenclature of indices, sets, parameters, and variables that have been used throughout this thesis.

Indices

i,j	Indices for element in modular product

Sets

Q	Set of interacting element pairs

Parameters

γ	Global scaling coefficient for C_3

Variables

N	Number of Variants
α_i	Element complexity of element i
β_{ij}	Interaction complexity of interaction between element i and j
A_{ij}	Connection of element i and j
f_{ij}	Interface complexity of interface between element i and j
E	Matrix energy

Terminology

Architecture	Arrangement and allocation of functionalities to physical components
Platform	Set of product elements arranged according to one or multiple product architectures
Modules	Self-contained, interchangeable units within TTI's architectures
Variants	Realized modular elements within TTI's platforms
Parts	Components that Variants are composed of
Interfaces	Defined connections between modules within TTI's platforms

Contents

List of Acronyms **ix**

Nomenclature **xi**

List of Figures **xvii**

List of Tables **xix**

1 Introduction **1**
1.1 Background . . . 1
1.2 Purpose . . . 2
1.3 Objectives . . . 2
1.4 Limitations . . . 3

2 Theory **5**
2.1 Modular product architecture . . . 5
2.2 Modules . . . 6
2.3 Parts . . . 6
2.4 Platforms . . . 7
2.5 Product classes . . . 7
2.6 Interfaces . . . 8
2.6.1 Generic Interface Component (GIC) . . . 9
2.6.2 GIC categories . . . 9
2.6.3 GIC classes . . . 9
2.6.4 SICs and KMLs . . . 10
2.7 KOLA . . . 10
2.7.1 EDB . . . 10
2.8 Adjacency matrix . . . 11
2.9 Matrix energy . . . 11
2.10 Design Structure Matrix . . . 11
2.11 Metrics . . . 12
2.11.1 Complexity . . . 12
2.11.2 Structural complexity . . . 13
2.11.3 Structural complexity metric . . . 13
2.11.4 Shannon entropy . . . 14
2.12 Weyuker's properties . . . 15

3 Methods **17**
3.1 Research approach . . . 17
3.1.1 Interview study on platform complexity . . . 17
3.1.2 Platform analysis . . . 18
3.1.3 Data collection . . . 19
3.1.3.1 Variant complexity . . . 19
3.1.3.2 GIC class complexity . . . 19
3.1.3.3 Extraction of Variant Connections . . . 20
3.1.3.4 Dataset . . . 20
3.1.4 Assessing Variant complexity . . . 21
3.1.5 Structural complexity metric . . . 22
3.2 Metric validation . . . 22
3.3 Case study . . . 23
3.4 Full-scale platform . . . 23
3.4.1 Automated data fetching . . . 23
3.4.2 Combining PCs . . . 24
3.5 Comparing Platforms . . . 24

4 Results **25**
4.1 Findings from interview study . . . 25
4.1.1 Platform complexity as interdependencies and system interactions 25
4.1.2 Variability and product diversity as key drivers . . . 25
4.1.3 Lack of standardization and unstable interfaces . . . 26
4.1.4 Architectural structure and modularization challenges . . . 26
4.1.5 Organizational and process-related factors . . . 26
4.1.6 Dynamic and evolving nature of complexity . . . 26
4.2 The proposed metric . . . 27
4.3 Results from platform analysis . . . 27
4.4 Simulated platform analysis . . . 27
4.5 Results from Weyuker validation . . . 29
4.6 Results from case study . . . 31
4.7 Results from platform comparison . . . 31
4.8 Results from automated data fetching . . . 31

5 Discussion **33**
5.1 Answer to research questions . . . 33
5.1.1 RQ1 . . . 33
5.1.2 RQ2 . . . 33
5.1.3 RQ3 . . . 34
5.2 Findings . . . 35
5.2.1 Interview study . . . 35
5.2.2 Platform comparison . . . 35
5.2.3 Complexity metric . . . 36
5.2.4 Simulated platform analysis and Shannon entropy . . . 37
5.2.5 Weyuker validation . . . 38
5.3 Reflections . . . 39

6 Conclusion **41**

Bibliography **43**

A Interview notes **I**
A.1 Interview questions I
A.2 Interview responses I
A.3 Closing remark III

B Simulation results **V**

List of Figures

2.1 Figure showing a simplified sketch of the Volvo modular platform and the configurability of truck variants, see Section 2.4. Modules connect via GIC interfaces, see Section 2.6. Each module consists of different Variants. Different combinations of Variants produce different truck configurations. 6
2.2 Figure describing the relationship between Parts and Variants. A Variant Family is shown containing three Variants. The Variants consist of different Part types, represented as colored shapes. Each Part type can contain multiple instances. 7
2.3 Figure showing an example of a Vehicle-PC and a Variant-PC. The colored shapes represent different Part types. Variants X, Y, and Z overlap in the PCs but contain different Part types depending on the PC. 8
2.4 (a) Binary DSM representing the presence or absence of interactions between system elements. (b) Multi-type DSM, in which each matrix entry is subdivided to represent different interaction types, with colors indicating the interaction category. For example, red = information exchange etc. 12

3.1 A DSM showing a the connections between a subset of Variants from the ICE platform. The rows and columns list the Variants in the subset. The crosses denote pairwise connections between the Variants. The Variant names have been removed to protect Volvo's integrity. . . 18
3.2 Figure explaining the process of combining Vehicle-PCs with Variant-PCs. The colored shapes represent different Parts. Matching Variants are merged together so that all Parts belonging to a Variant are accounted for. Only Variants that match in both PCs are merged. Overlapping Parts are ignored. 24

4.1 Bar chart showing the number of modules, Variants, SICs, and Parts alongside the proposed metric $C_{proposed}$ and simpler metric C_{simple} for six simulated platforms with randomly sampled parameter values. The number of Parts is scaled to 20% of its original value for readability. 28

B.1 Bar chart showing the number of modules, Variants, SICs, and Parts alongside the proposed metric $C_{proposed}$ and simpler metric C_{simple} for six simulated platforms with randomly sampled parameter values. The number of Parts is scaled to 20% of its original value for readability. V
B.2 Bar chart showing the number of modules, Variants, SICs, and Parts alongside the proposed metric $C_{proposed}$ and simpler metric C_{simple} for six simulated platforms with randomly sampled parameter values. The number of Parts is scaled to 20% of its original value for readability. VI
B.3 Bar chart showing the number of modules, Variants, SICs, and Parts alongside the proposed metric $C_{proposed}$ and simpler metric C_{simple} for six simulated platforms with randomly sampled parameter values. The number of Parts is scaled to 20% of its original value for readability. VII
B.4 Bar chart showing the number of modules, Variants, SICs, and Parts alongside the proposed metric $C_{proposed}$ and simpler metric C_{simple} for six simulated platforms with randomly sampled parameter values. The number of Parts is scaled to 20% of its original value for readability. VIII

List of Tables

2.1 Table showing the different GIC classes. The classes are either of a media-carrying, mechanical, or geometric nature. 10

3.1 Table showing the complexity ranking of each GIC class. The left column lists the classes and the right column lists the assigned complexity scores. This ranking is based on Volvo's A-interface importance hierarchy and the results from the interview study. 20

4.1 Table showing the structural complexity values of the sub-platform investigated in the case study. The values are rounded to 4 significant figures. 31

4.2 Table showing the structural complexity values of the ICE and BEV platforms. The values are rounded to 4 significant figures. 31

1

Introduction

This thesis investigates architectural complexity. Specifically, structural complexity of modular vehicle platforms. In this report, a platform is defined as a set of product elements that are arranged according to a product architecture. A platform can contain multiple architectures. In addition, throughout this thesis, a product configuration is defined as a platform instance. This first chapter describes the background, purpose, objective, and limitations of this thesis.

1.1 Background

Production efficiency is vital for companies in a competitive landscape. Many modern companies utilize advanced production techniques that are more flexible than mass production of structurally homogeneous products. One of these companies is the Trucks Technology & Industrial Division within Volvo Group (TTI), which develops modular trucks for scalable transportation solutions.

A modular approach is attractive to companies because it is cost-effective, enabling advantages such as greater product variety and flexibility. Product variety is necessary to satisfy customers in a wider spectrum of demand, which can lead to gains in market share [1]. Flexibility allows for efficient future development of the product, without the need to reinvent the product at every step [2].

Modular products are designed to allow configuration and reconfiguration through interchangeable components, either at or after purchase. The products are thus divided into different segments called modules [3]. A module consists of a set of module variants that can be used in a configuration of the product. It is important to note that, for most modules, only one of its variants can be used in a configuration [4]. For example, if the product is a truck that has a module encapsulating the engine, then only one variant of the engine module, say a 13-liter or 17-liter engine, can be used in any realized truck configuration. Furthermore, depending on the architecture of the product, i.e. how the partition of modules is designed, module variants can have their own modules and parts.

A key concept in this context is complexity, specifically architectural complexity, which can be defined as the number of elements in a system and the extent of their interdependencies [5]. The disadvantage with a modular approach is that large numbers of modules increase the complexity of the platform and the module interfaces [6]. A high level of complexity will decrease development efficiency and

maintainability of the product. This phenomenon arises because each module is required to meet the constraints of every other module it shares an interface with [7].

While modularity introduces complexity through interfaces and interdependencies, the alternative monolithic approach is often more complex. In product-lines with many product variants, non-modular designs tend to be tightly coupled, meaning that a change in one component often requires changes in others. As a result, each product variant may involve substantial redesign, which increases the overall complexity. Modularity is therefore used as a way to manage this complexity by standardizing interfaces and limiting the impact of changes to individual modules. However, the outcome depends on how well the architecture is designed. Poor modularization can introduce unnecessary dependencies and reduce these benefits [8]. In order to optimize modularization, balancing module quantity with complexity is required [5]. Complexity can thus be used as a way to assess and improve the architecture.

An unnecessarily complex modular architecture will require more effort to be developed and updated than a well modularized architecture [9]. Especially, larger products with longer development life cycles benefit from a well designed architecture. With poor architecture design, these types of products might become exceedingly expensive to manage due to increased development time. Development costs are thus connected to platform complexity [5] [10]. Furthermore, as stated by Chávez and Luis [11], added complexity does not necessarily result in larger revenue and will most certainly increase overall production cost. It follows that to generate maximum profit, a balance between complexity and modular variety is essential. To quantify and process this complexity, metrics and methods for analysis will be developed.

1.2 Purpose

The purpose of this thesis is to analyse the complexity of modular vehicle platforms in collaboration with TTI's Platform and Architecture team within the CAST organization. The study aims to develop a metric for assessing the complexity of modular vehicle platforms, analyse platform complexity, and identify the principal drivers contributing to increased architectural complexity. The metric is intended to measure the architectural complexity of different platform designs in order to enable objective comparisons of different designs.

1.3 Objectives

The thesis aims to develop a data-driven complexity metric as a way to analyse TTI's modular platform architecture. To develop and adapt the metric, existing metrics will be explored and TTI's platforms will be analysed. The analysis is expected to provide a deeper understanding of the characteristics of the platforms. The documented module and interface variations provided by TTI will form the basis for the metric. Additionally, effective and reliable ways of handling TTI's product data will be explored in order to improve future data-driven analyses.

Below are the research questions (RQs) this thesis will explore.

- RQ1: How can module and interface variation be systematically measured to quantify architectural complexity in a platform?
- RQ2: How can Volvo TTI's data fetching be structured to enable efficient and reliable data-driven analyses?
- RQ3: What complexity metrics can provide insight into the structural characteristics of modular vehicle platforms?

1.4 Limitations

For this thesis, direct API calls to TTI's database were not possible due to restrictions on access. The reason for the restriction was that the project was, according to Volvo, not extensive enough to make up for the security risks implied by direct access. Instead, all data was fetched from one of TTI's user tools called Engineering Database (EDB).

Due to the broad scope of complexity, not all areas of complexity were covered. This thesis focuses on one definition of structural complexity because it is relatively quantifiable with respect to objectivity. Other areas of complexity with large impacts on TTI's profitability, such as organizational complexity and documentation complexity are not in focus. These areas are connected to other challenges in the company. However, they are of a more subjective nature than structural complexity and demand a different approach to the one conducted in this thesis.

Because there is no well established unit of structural complexity, the metrics proposed in this thesis are of a relative nature. In other words, the metric values of a platform are only useful in comparisons with other platforms.

TTI has different interface categories that dictate the scope of the interfaces. There are A-,B-, and C-interfaces. In this thesis, only A-interfaces are taken into account when assessing the platform's structural complexity. This is because they have the highest strategic importance for the platform as a whole. Furthermore, they are a minority among the total set of interfaces and are strictly regulated, while still being abundant enough to make a solid foundation for a data driven assessment of complexity. Thus, a selection of only A-interfaces best captures the principal components of complexity while reducing noise from bulks of unstable B- and C-interfaces.

A further limitation concerns the scope of analysis. This thesis only focuses on the hardware of products. There are two reasons for this, firstly because the database supplied by TTI is based on hardware, and secondly because of a limited time frame. There are notable similarities between modular hardware and modular software. A well-defined scope, clear-cut modules, data classes, and interfaces are common objectives in both domains. In addition, hardware and software tend to be highly interconnected in modern vehicles. Given these common objectives and the high interplay between hardware and software, analysis of both domains provides a holistic perspective. However, software development and deployment are usually more rapid

processes than hardware development and deployment. This is because a software update can be developed and installed over the course of days or even hours, while hardware updates can take months or years. This makes the two domains separate from some perspectives, suggesting that hardware-focused analysis yields insights that a cross-domain study could obscure.

2

Theory

The following chapter describes the theoretical concepts used throughout this thesis. It gives the foundation required to comprehend the subject of this thesis.

2.1 Modular product architecture

According to Ulrich [12], product architecture is the scheme by which the function of a product is allocated to physical components. More precisely, it consists of the arrangement of functional elements, the mapping from functional elements to physical components, and the specification of the interfaces among interacting components. In contrast to integrated product architecture, where functions and components are tightly coupled, modular product architecture partitions a product into self-contained units, called modules, that have clear functional purposes and interact only through standardized interfaces [1].

A modular approach is effective for products that are designed to adapt to many specific functionalities, and conversely, to reuse common elements in different product variations [13]. Firstly, a product that can be adapted to many functionalities is competitive on the market, as it has the ability to satisfy a wider set of customer needs [1]. Secondly, from a design perspective, it is efficient to reuse elements that fulfill the same purpose in many products. Reuse saves time and effort that would otherwise be spent designing different elements that serve the same purpose [14][15][16]. Thus, the standardization of elements used in many variants of the product is key in an efficient modular architecture.

In order for these common elements to fit into different product variants, they need to be designed with all considered product variants in mind. When designing a product with multiple performance steps, such as a truck, standardization should be limited to the set of elements that does not include performance steps. This is because a standardization of a module that contains performance steps will fix the intended functionality of the module to a particular performance step. This then leads to the product being unsuitable for a subset of customer needs. The optimal solution for any modular architecture is one that balances reusability and functional adaptability [13]. Well defined modules are thus a necessity for any efficient solution. Figure 2.1 shows a simplified sketch explaining the relationship between architecture, platform, modules, and truck configurations.

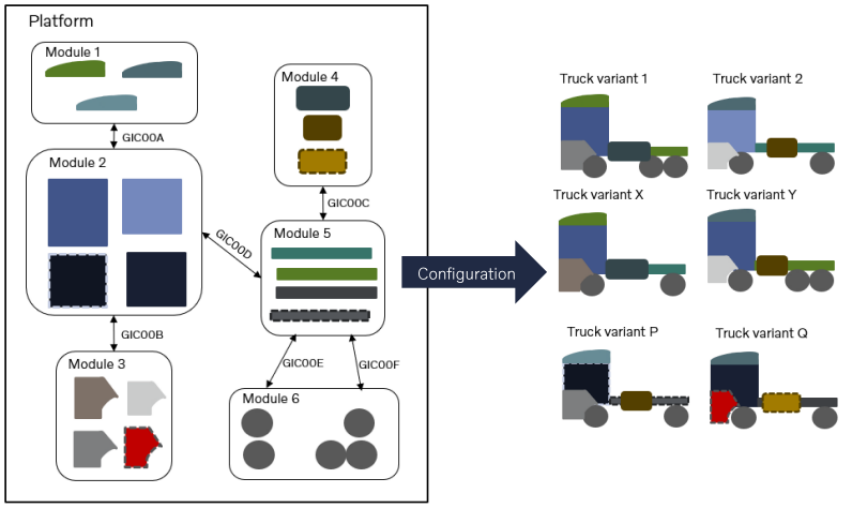

Figure 2.1: Figure showing a simplified sketch of the Volvo modular platform and the configurability of truck variants, see Section 2.4. Modules connect via GIC interfaces, see Section 2.6. Each module consists of different Variants. Different combinations of Variants produce different truck configurations.

2.2 Modules

Modules can be designed in many different fashions depending on the objective of the product [13]. Thus, a product line can be modularized in multiple ways, since different modular architectures, each reflecting different design trade-offs, can exist for the same product line. Products can be modularized in multiple layers. A module can therefore contain submodules, the submodules can in turn contain submodules of their own, and so forth [17]. Because of this, the modularization task can be complex and require extensive analysis before an efficient architecture can be developed.

As explained in Section 2.1, modules contain different variants of elements that can offer either different functionalities, or performance steps of the same functionality. In this thesis, we will focus on one layer of modules of the TTI modular truck. The realized module variants and modules are named *Variants* and *Variant Families*, respectively [4]. Henceforth, these names will be used. Examples of Variants and Variant Families are presented in Figure 2.1.

2.3 Parts

Variants in TTI's modular truck architecture are designed to fulfill specific customer needs and are usually of different performance steps [4]. For example, the engine Variant Family has a selection of different engine performance steps. Each Variant offers its own commercial advantage, from the cost efficiency of a smaller engine to the performance capability of a larger one. The Variants are composed of *Parts*. Variants typically consist of multiple Part types, each of which may appear more than once within the Variant. The layout of Variants and Parts is described in Figure 2.2.

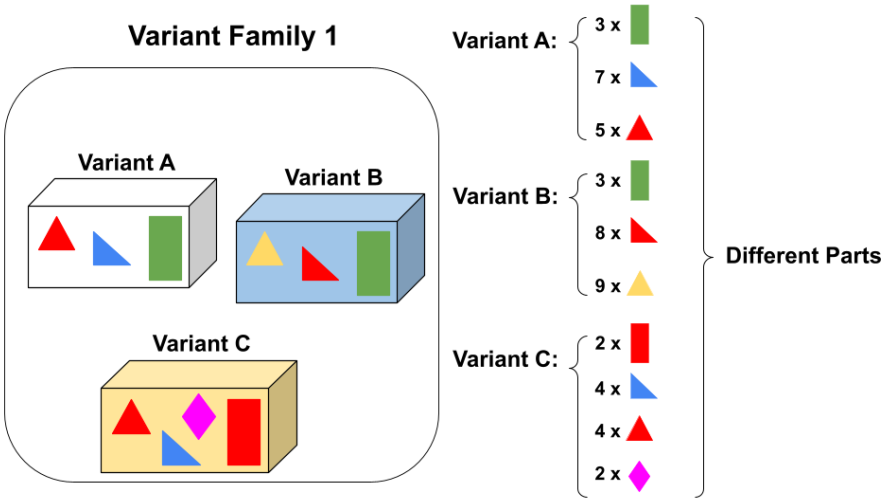

Figure 2.2: Figure describing the relationship between Parts and Variants. A Variant Family is shown containing three Variants. The Variants consist of different Part types, represented as colored shapes. Each Part type can contain multiple instances.

2.4 Platforms

A product platform can be defined as a collection of shared assets that serve as a common basis for a set of related products. These assets may include components, production processes, knowledge, and organizational relationships. By building multiple product variants from the same platform, firms can achieve both efficiency through commonality and differentiation through product-specific variation [18]. In this thesis, a platform is defined as a set of product elements that adhere to one or multiple product architectures. The TTI modular platform is thus the set of actual Variants that can be combined into a truck according to the modular architecture. A platform can be divided into sub-platforms for the sake of analysis.

2.5 Product classes

TTI's products are categorized into different product classes (PC), each containing their own product type, e.g. a medium- or heavy-duty truck. These PCs that contain a vehicle type will be called *Vehicle-PCs* in this thesis. There are also product classes for some of the more significant Variant Families, such as the engine Variant Family. These PCs will be called *Variant-PCs*. These Variant-PCs have no modules but encapsulate all Variants in their corresponding Variant Family. Variants in Variant-PCs generally have a higher number of Parts than Variants in Vehicle-PCs. By incorporating these significant Variant Families into their own PC, they can be more easily managed, especially since they can be shared by multiple Vehicle-PCs. However, Variants of Variant-PCs can still be included in Vehicle-PCs. Because of this, there exists some overlap in the documentation of these Variants. Variant-PCs and Vehicle-PCs are described graphically in Figure 2.3.

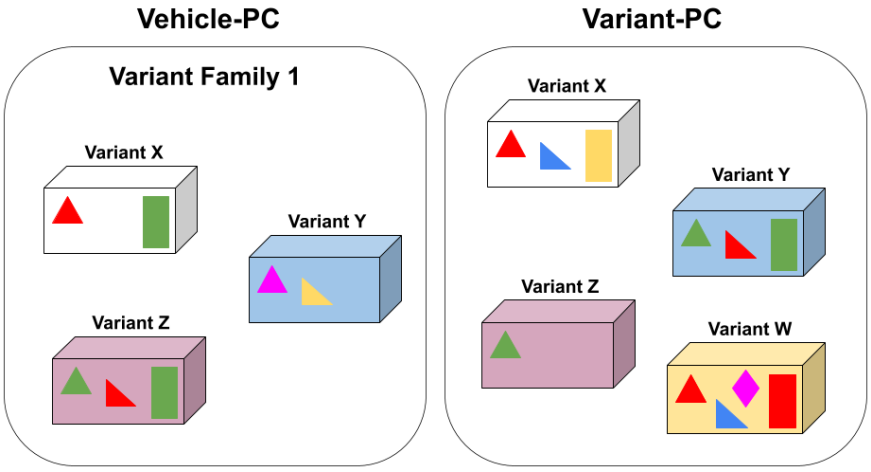

Figure 2.3: Figure showing an example of a Vehicle-PC and a Variant-PC. The colored shapes represent different Part types. Variants X, Y, and Z overlap in the PCs but contain different Part types depending on the PC.

In this thesis, a PC is viewed as a member of a platform, e.g. multiple heavy-duty truck PCs can belong to the same platform as long as the trucks share some overarching trait. The Vehicle-PCs analysed in this thesis are two internal combustion engine (ICE) truck PCs and two battery electric vehicle (BEV) truck PCs. The ICE PCs and the BEV PCs are combined into an ICE platform and a BEV platform, respectively. The Variant-PCs analysed in this thesis are the engine, gearbox, front axle, rear axle, and cab PCs. These Variant-PCs are analysed because they have Variants that are included in the ICE- and BEV PCs.

2.6 Interfaces

Variant Families in a platform connect to each other via interfaces. In a product line where an existing Variant is updated to a new performance level, or replaced by a newer version, the corresponding Variant Family must be adapted to fit the existing interface, or the interface must be adjusted accordingly. However, when a Variant Family consists of multiple Variants, it is often more efficient to adapt the changed Variant to the existing interface than to modify the interface and consequently adjust several surrounding Variants. This keeps the change localized and reduces complexity in the long term.

The difficulty in designing an efficient platform is to define interfaces that maximize compatibility of Variants. It may be sufficient to optimize the interfaces in accordance with the current sets of Variants, but this may limit the scalability of the platform because new Variant versions with different requirements might be added post-optimization. For scalable solutions, continuous interface updates can solve this problem.

2.6.1 Generic Interface Component (GIC)

Volvo uses the term Generic Interface Component (GIC), which describes the type of interface, including its general characteristics and intended functionality without specifying any particular implementation. For instance, a geometric GIC can define a coordinate system in which physical connections between Variants exists. Examples of GICs are shown in Figure 2.1.

2.6.2 GIC categories

GICs are split into different categories based on their strategic importance. There are A-,B- and C-interfaces. C-interfaces are defined within modules. They are only regulated by the organization responsible for their respective module. B-interfaces are defined between modules and are thus subject to a larger range of Variants. These interfaces are regulated by the organizations responsible for the modules that the interfaces affect [4].

A-interfaces are a collection of the most important B- and C-interfaces from a macro platform perspective. This is based on, but not limited to, whether the interface: i) is defining basic vehicle architecture, ii) has high complexity or high risk for change propagation, iii) has high-cost impact to change, develop or maintain variation [4]. A-interfaces are regulated by the Platform and Architecture team, with a holistic approach in mind. This is because an incompatible A-interface renders many modules useless due to the high interdependency of these interfaces. Only changes that are completely compatible to all affected modules are approved by the Platform and Architecture team [4].

This thesis focuses on A-interfaces because they are the most important from a platform complexity perspective. They capture a majority of the interdependencies between modules, as well as covering the major segments of the truck. Important to the complexity assessment, A-interfaces are tightly regulated and all changes to them have to go via the Platform and Architecture team. This enforces that the data used for platform complexity assessment is high quality with a lack of noise from smaller interfaces [4].

2.6.3 GIC classes

Each GIC also belongs to one or more classes. These classes describe the nature of an interface, whether it is media-carrying, mechanical, or geometric, as well as its specific purpose [4]. Media-carrying interfaces connect Variants that share informational flow. Mechanical interfaces connect Variants through physical linkage. Geometric interfaces dictate positions or geometrical boundaries of Variants. A GIC can belong to multiple classes. The classes have different impacts on the interdependency of the interface. Thus, the class is connected to the complexity of the interface. For instance, many Variants are dependent on the Master Reference-class, leading to that class being more complex than the relatively local Position-class. All classes are listed in Table 2.1.

GIC Class	Nature
Master Reference	Geometric
System Platform	Geometric
Vehicle Architecture Position	Geometric
Vehicle Platform Dimension	Geometric
Layout Position	Geometric
Mechanical contact (Layout impact)	Geometric
Boundary	Geometric
Position	Geometric
Mechanical contact	Mechanical
Media contact	Media-carrying

Table 2.1: Table showing the different GIC classes. The classes are either of a media-carrying, mechanical, or geometric nature.

2.6.4 SICs and KMLs

Volvo denotes a Specific Interface Component (SIC) to be a concrete realization of an interface for a specific use case. For a geometric interface defined by a GIC, the SIC may specify the exact X, Y, and Z coordinates of the physical connection between the Variants. For example, the interface could be defined at $(x, y, z) = (10, 25, 4)$, illustrating where the Variants meet in practice.

A KOLA Module List (KML) defines the set of Variant selections associated with a given SIC. The list includes no more than one Variant per Variant Family and serves as Volvo's standard representation of the SIC's Variant combination [4].

2.7 KOLA

The TTI organization within Volvo Group utilizes a Product Data Management (PDM) system named KOLA, introduced in the 1970s [4]. KOLA is a software application with a graphical user interface (GUI) and contains data regarding all Parts, Variants, and interfaces that are in production and development. KOLA has a dataset consisting of A-interfaces and their respective SICs. Via a KML, each SIC lists which Variants are connected through it. In other words, this list of Variants constitutes one configuration of Variants that is compatible with the corresponding GIC. By fetching module and interface data from KOLA and applying a suitable metric, an objective complexity measure of platforms is possible.

2.7.1 EDB

EDB is a Volvo-developed framework and web-based user-interface that is used to access technical product data [4]. It functions as a data warehouse by collecting and organizing information from multiple sources, such as KOLA, and providing links to external systems when needed. Multiple tools exist in EDB. These are used

to support the retrieval, management, and navigation of technical data related to vehicle products. EDB is the main source of product data used in this thesis.

2.8 Adjacency matrix

An adjacency matrix is a square matrix used to represent a finite graph and its interconnections. It consists of a grid where both rows and columns are nodes (vertices) of a graph. In its simplest form, the entries are binary; an entry of one indicates a connection between the two nodes, while a zero indicates no connection. A graph with n nodes will have an adjacency matrix of size $n \times n$.

2.9 Matrix energy

In this thesis, the product architecture is represented by a binary adjacency matrix **A**, where rows and columns correspond to Variants, and each element indicates the presence or absence of an interface between two Variants:

$$a_{ij} = \begin{cases} 1, & \text{if there is an interface between Variant } i \text{ and Variant } j \\ 0, & \text{otherwise} \end{cases} \tag{2.1}$$

The matrix energy is then defined as a scalar measure of the overall level of connectivity in the platform, computed as the sum of all elements in the adjacency matrix:

$$E(A) = \sum_i \sum_j a_{ij} \tag{2.2}$$

2.10 Design Structure Matrix

In engineering design, different types of adjacency matrices are used to analyse the structure of systems. Particularly, the Design Structure Matrix (DSM) is used for product architecture analysis [19]. A DSM is a tool used to model and visualize the interactions between elements in a complex system [20]. The elements can represent, for example, components, submodules, or modules, depending on the level of abstraction in the system architecture. In a DSM, each row and column correspond to a system element, and the intersection between a row and a column indicates an interaction or dependency between the corresponding elements. The result is a square $N \times N$ matrix that maps the interactions among the N system elements, similar to an adjacency matrix [19].

In complex systems, these interactions are often multi-dimensional, apart from only mechanical connections, also involving various types of exchanges such as mass, energy, or information. To capture this complexity, a multidimensional DSM can be constructed. However, such a matrix can be hard to visualize for higher dimensions.

To represent a multidimensional DSM in two dimensions, a 2D-DSM can be extended by collapsing the multidimensional DSM so that each intersection becomes a subgrid in the 2D-DSM. Each cell within the sub-grid represents a specific interaction type. However, in this thesis only binary 2D-DSMs are considered, i.e. all interactions are grouped into one type and are represented by either a marked or unmarked cell. Examples of binary and multi-type DSMs are shown in Figure 2.4. The diagonal of the DSM is typically empty (blacked out in this case), since self-interactions are generally not considered [21].

	E1	E2	E3	E4	E5
E1			x		
E2				x	
E3	x				
E4		x			
E5					

(a) Binary DSM

	E1	E2	E3	E4	E5
E1		x x			
E2	x x			x x	
E3					
E4		x x			
E5					

(b) Multi-type DSM

Figure 2.4: (a) Binary DSM representing the presence or absence of interactions between system elements. (b) Multi-type DSM, in which each matrix entry is subdivided to represent different interaction types, with colors indicating the interaction category. For example, red = information exchange etc.

2.11 Metrics

Analysing existing platforms can improve platform development. By investigating platform complexity, quantifiable metrics can be developed, which in turn can be used to support more informed design decisions and more efficient platforms.

2.11.1 Complexity

In this section, complexity will be defined. Complexity is a multifaceted concept, and there is no single universally accepted definition. Prior studies suggest that complexity can be measured through functional decomposition, as in Bashir and Thomson [22]. Suh [23] describes complexity in a product as the probability of successfully fulfilling the functional requirements of the product. The author also proposes metrics for measuring this probability. Some studies measure complexity from an economic performance perspective. This view is applied to modular platforms in Schuh et al. [24]. Other studies focus more on the inherent traits of the product. Simon [25] describes a complex system as one made up of a large number of parts that interact in a non-simple way, and argues that complexity frequently takes the form

of hierarchy. Serrat [26] similarly describes complexity as arising from interacting elements, where the behaviour of the whole cannot be fully understood by studying the elements in isolation.

2.11.2 Structural complexity

Lanza et al. [27] defines structural complexity as follows: "Structural Complexity, a subset of system complexity, is particularly pertinent in engineered systems as it directly relates to the architecture, quantity, and heterogeneity of system elements and their interconnections", a definition supported by de Weck et al. [28]. Summers and Shah [29] compares different complexity metrics and proposes additional metrics based on size, coupling, and solvability with respect to design, problem, process, and product. In the context of integrated architecture, Raja et al. [30] proposes a simulation-assisted complexity metric for systematically and quantitatively assessing the structural complexity in aero-engine structures. The authors state that such a metric can "assist the design and optimization processes tremendously" and that the metric "can be used to compare design alternatives" [30]. Similarly, Sinha and de Weck [31] takes a structurally focused approach to complexity and proposes a metric that has become well established in the field of structural complexity. In product platform research, complexity is also treated structurally through the number of shared physical elements, parts, and processes in a product family [18]. In this thesis, structural complexity is used because the focus is on Volvo TTI's platforms, specifically the number of modules and system elements, as well as the interfaces between them. This serves as a means of comparing alternative platform designs and supporting the design and optimization process, in line with the application proposed by Raja et al. [30].

2.11.3 Structural complexity metric

Sinha and de Weck [31] proposed a formula for estimating the structural complexity (SC) of a system, shown in Equation (2.3).

$$\begin{aligned} SC &= C_1 + C_2 C_3 \\ &= \sum_{i=1}^{N} \alpha_i + \left[\sum_{i=1}^{N} \sum_{j=1}^{N} \beta_{ij} A_{ij} \right] \gamma E(A) \end{aligned} \tag{2.3}$$

The SC metric is composed of two main contributions: the first being complexity of individual elements, C_1, and the second being the contribution from inter-element interactions, $C_2 C_3$.

The element complexity is given by C_1, where:

$$C_1 = \sum_{i=1}^{N} \alpha_i \tag{2.4}$$

Here, α_i denotes the complexity of element i, in the form of a scalar value. Individual elements may exhibit varying levels of complexity due to factors such as intricate design, counter-intuitive behaviour, physical constraints or interaction with other components. In the absence of interconnections, the structural complexity reduces to the sum of the individual element complexities.

The interaction contribution is given by C_2C_3:

$$C_2 = \sum_{i=1}^{N} \sum_{j=1}^{N} \beta_{ij} A_{ij} \tag{2.5}$$

$$C_3 = \gamma E(A) \tag{2.6}$$

where γ is a global scaling factor and $E(A)$ is the matrix energy of the system's adjacency matrix A, described in Section 2.9. The interaction contribution captures the inter-element interactions, C_2C_3. Each interaction is weighted according to the type of interface between the two elements, as proposed by Sinha and de Weck [31].

The interaction weight β_{ij} is defined as:

$$\beta_{ij} = f_{ij}\alpha_i\alpha_j \tag{2.7}$$

where:

- α_i and α_j denote the complexities of elements i and j, respectively.
- f_{ij} represents the complexity of the interaction between two elements, which often depends on the nature of the interaction, e.g. mechanical, electrical, informational, or energy-based connections.

The adjacency matrix A_{ij}, as described in Section 2.8, defines whether two elements are connected:

$$A_{ij} = \begin{cases} 1, & \text{if } i \neq j \text{ and } (i,j) \in Q \\ 0, & \text{otherwise} \end{cases} \tag{2.8}$$

where Q represents the set of valid connections in the system. The DSM in Figure 2.4a provides an illustrative example of such an adjacency matrix, where elements E1-E5 represent a generic system used to visualize the connectivity structure.

2.11.4 Shannon entropy

In this work, Shannon entropy is used to derive the Variant complexity α, which is subsequently combined with the SC metric, Equation (2.3). Shannon entropy, originally introduced by Shannon [32], is a quantitative measure of uncertainty within a probability distribution. It is adapted in this work as it provides a consistent and quantitative measure of uncertainty and diversity in the distribution of Parts in a Variant, which can be interpreted as a measure of Variant-level structural complexity.

By evaluating a discrete set of probabilities, it captures the degree of dispersion or concentration among the Part types of a Variant [32]. The total number of Parts in a Variant has no significance to the Variant complexity, other than changing the granularity of the probability distribution. It can be interpreted as the distinction between demanding and complex. A low entropy value indicates a concentrated and thus highly predictable distribution, whereas a high entropy value reflects a more evenly distributed and therefore more uncertain system structure. Shannon entropy of a discrete random variable X with possible outcomes x_i and the corresponding probabilities $p(x_i)$ is defined as:

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log_2(p(x_i)) \tag{2.9}$$

where n denotes the number of distinct states and $p(x_i)$ represents the probability of occurrence of state x_i. In the context of a platform, a state corresponds to a Part type within a given Variant. The entropy $H(X)$ is computed separately for each Variant based on the distribution of Part types within that Variant across the platform. The resulting entropy value is used as a measure of Variant complexity.

2.12 Weyuker's properties

Weyuker's properties, proposed by Weyuker [33], consist of nine fundamental properties used to evaluate the effectiveness and validity of software complexity measures. It was originally developed in the context of software systems, however these properties are broadly applicable to complexity metrics defined over structured systems. In this work, Weyuker's properties are considered as a qualitative framework for evaluating the behaviour of metrics when applied to modular vehicle platforms. It is relevant in this context, since both software systems and modular product architectures can be described as collections of interconnected elements and interfaces. Let A and B denote two modular vehicle platforms, or sub-platforms, and let $SC(A)$ represent the structural complexity of platform A. The properties can be interpreted as follows in the context of modular vehicle architecture:

1. There exist vehicle platforms A and B whose structural complexity differs, i.e. $SC(A) \neq SC(B)$.
2. For any given complexity value c, only a finite number of vehicle platforms can have that value.
3. Distinct platforms A and B may have the same structural complexity, i.e. $SC(A) = SC(B)$.
4. Two platforms that provide the same functional capabilities may still differ in structural complexity, i.e. $SC(A) \neq SC(B)$.
5. For any two sub-platforms A and B, the complexity of the combined platform is greater than or equal to the complexity of each individual platform, i.e. $SC(A) \leq SC(A \cup B)$ and $SC(B) \leq SC(A \cup B)$.

6. There exist platforms A, B, and O such that $SC(A) = SC(B)$, but when combined with platform O, the resulting complexities differ, i.e. $SC(A \cup O) \neq SC(B \cup O)$.
7. Reordering modules within a platform may change its structural complexity. That is, there exist platforms A and B, where B is a permutation of A, such that $SC(A) \neq SC(B)$.
8. Renaming or relabelling modules does not affect the structural complexity, i.e. if A is a renaming of B, then $SC(A) = SC(B)$.
9. The complexity of a combined sub-platform may exceed the sum of the individual complexities, i.e. there exist A and B such that $SC(A) + SC(B) < SC(A \cup B)$.

3

Methods

This thesis explores methods for developing complexity metrics for modular platform architecture. To address the research questions outlined in Chapter 1, a systematic and iterative research approach was conducted.

3.1 Research approach

First, an understanding of complexity and its various forms was developed through a thorough literature review. This review focused on i) definitions of complexity, e.g. how authors define complexity in different areas ii) existing complexity metrics in engineering and related fields, and iii) approaches to analysing modular and platform-based architectures, the theoretical background and reviewed metrics are presented in Section 2.11.

Following the literature review, Sinha and de Weck's structural complexity formula was selected as the primary framework for quantifying architectural complexity. This framework was chosen because it captures both structural and behavioral aspects of a system by incorporating three key components: (i) element complexity, (ii) interaction complexity, and (iii) connectivity. As a result, it provides a more holistic representation of architectural complexity than metrics that focus solely on component counts or connectivity. The SC formulation is particularly suitable for modular vehicle architecture, where both the arrangement of modules and the pattern of interfaces across Variants are major drivers of complexity.

3.1.1 Interview study on platform complexity

To complement the theoretical perspective on platform complexity, interviews were conducted with practitioners within TTI's Platform and Architecture team. The purpose of this study was to capture how platform complexity is understood and experienced in practice, and to identify the factors that contribute to it and its impact on daily work.

Data was collected through semi-structured interviews focusing on how platform complexity is defined, which factors are perceived as driving complexity, and how such complexity affects daily work. The complete list of interview questions is provided in Appendix A. The participants consisted of practitioners with different roles and responsibilities, which ensured a range of perspectives. Particular attention

was given to frequently mentioned aspects such as system interactions, dependencies, variability, interface standardization, and organizational factors (e.g. ways of working and documentation practices).

The responses were analysed to identify recurring themes and patterns. These insights also served as a guiding foundation for the development and selection of the metric used to capture platform complexity.

3.1.2 Platform analysis

In order to grasp the layout of TTI's platforms, a thorough analysis of the platforms' different segments was conducted. This was necessary to formulate a method for developing and applying a metric to the platforms. The analysis included constructing and interpreting DSMs for the ICE and BEV platforms. An example of a DSM used for analysing the ICE platform is shown in Figure 3.1, where the Variant names have been removed to protect Volvo's integrity. The GIC classes, explained in Section 2.6.3, were also investigated. By downloading and inspecting GIC and Variant files, a comprehensive understanding of the platforms' segments was achieved. A method for assessing platform complexity was devised based on this insight, the literature study in Section 2.11.1, and the interview study in Section 3.1.1.

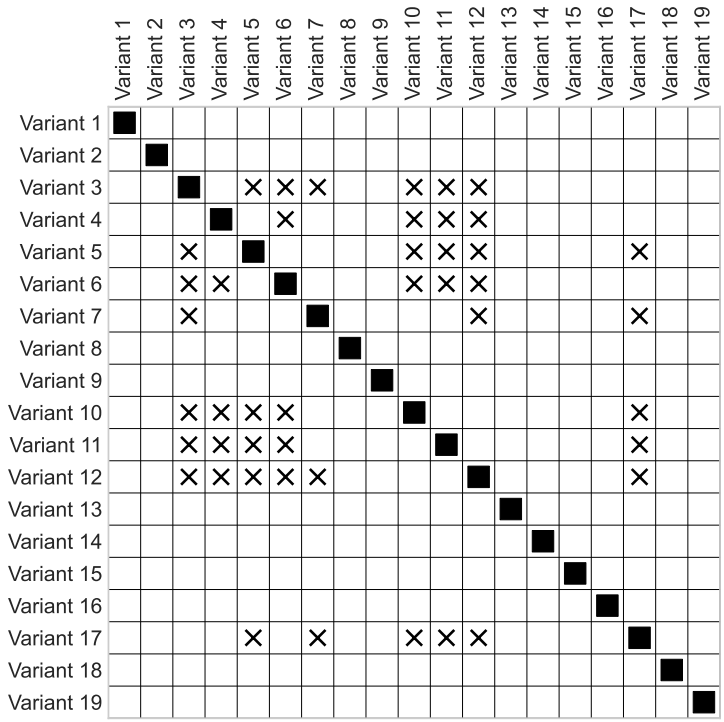

Figure 3.1: A DSM showing a the connections between a subset of Variants from the ICE platform. The rows and columns list the Variants in the subset. The crosses denote pairwise connections between the Variants. The Variant names have been removed to protect Volvo's integrity.

3.1.3 Data collection

Assessing the structural complexity of the platform requires a systematic approach to data collection. For this, various tools of EDB were used.

3.1.3.1 Variant complexity

As previously stated in Section 2.11.4, the Parts for each Variant within the platform are used to assess the Variant complexity, which is modelled as the element complexity α in Sinha and de Weck's SC formula [31], see Equation (2.3). EDB has a tool which, given a Variant and a PC, returns a downloadable "CSV"-file that contains the names of Parts used for that Variant. The tool is provided as a web form, requiring manual entry of each Variant followed by submission via a search function. The search generates a set of downloadable files, albeit with inconsistent naming conventions, which were used to calculate the Shannon entropy of each Variant.

3.1.3.2 GIC class complexity

As explained and listed in Section 2.6.3 and Table 2.1 respectively, GICs are divided into classes that differ in interdependency and thus complexity. Because of this, interactions between Variants have different complexity depending on the interface class. This is modelled in Sinha and de Weck's SC formula [31] as the interaction complexity f, see Equation (2.7). Based on internal Volvo documentation [4], the interface classes, specifically for A-interfaces, have different importance from a platform perspective. In the documentation, classes are organized in a hierarchy of importance. Based on this hierarchy and the results from the interview study, a complexity ranking of GIC classes was made, shown in Table 3.1. The ranking distinguished different GIC complexity levels. The values 1–4 were chosen for their simplicity, which facilitated the analysis of interaction complexity across large-scale platforms.

Due to GICs being able to belong to multiple classes, as explained in Section 2.6.3, the highest complexity score of all associated classes is assigned to each GIC. For example, if GIC001 belongs to the classes: System Platform, Layout Position and Boundary, it will be assigned a complexity value of 3.

GIC Class	Complexity score (f)
Master Reference	4
System Platform	3
Vehicle Architecture Position	3
Vehicle Platform Dimension	3
Layout Position	2
Mechanical contact (Layout impact)	2
Boundary	1
Position	1
Mechanical contact	1
Media contact	1

Table 3.1: Table showing the complexity ranking of each GIC class. The left column lists the classes and the right column lists the assigned complexity scores. This ranking is based on Volvo's A-interface importance hierarchy and the results from the interview study.

3.1.3.3 Extraction of Variant Connections

To calculate a platform's C_2 and C_3 values, information describing the connections between Variants within the platform was required. These connections are represented by the interface relationships between Variants that together define the platform. This information was retrieved from EDB using a tool which, given a GIC, downloads an ".xlsx" file that contains every SIC associated with the given GIC. For each SIC, the corresponding KML was extracted using Python and stored in a list. Because the SC formula counts interactions in pairs, modelled as the adjacency matrix shown in Equation (2.8), the KMLs had to be separated into combinations of size two. As a KML typically contains more than two Variants, all pairwise connections between Variants were generated and stored in a CSV-file. Furthermore, since a platform may include multiple GICs between Variants, the GIC associated with each pairwise connection was explicitly documented. This enabled the calculation of the platform's C_2 value by evaluating each connection pair and the complexity of their shared interface, determined using the GIC class ranking in Table 3.1. From the pairwise connections, the matrix energy was calculated and multiplied with γ, producing C_3 according to Equation (2.6). A γ-value of 0.1 was chosen to balance the Variant complexity and interaction complexity contributions. This was based on parameter tuning on the full-scale platform described later in Section 3.4.

3.1.3.4 Dataset

When fetching the connected Parts data for all unique Variants in the KMLs, there were a few mismatches in the dataset. For some Variants, some connected Parts files were missing in EDB. This is most likely due to the documentation of those Variants being in progress, a result of continuous development of the platforms. To solve this issue, empty connected Parts files were created and added to the connected Parts data set.

3.1.4 Assessing Variant complexity

In order to quantify Variant complexity α, Shannon entropy, see Section 2.11.4, was applied to the Parts of each Variant within the platform. For every Variant, a discrete probability distribution was constructed based on the relative frequency of Part types. This was a suitable measure because Shannon entropy described how varied the Part types were within a Variant, which assessed the structural complexity without deeper knowledge of the Variant's structure. Variant-level structural complexity could be achieved with higher precision by rigorously studying each Variant's physical structure using metrics for integrated architectures, as in Raja et al. [30]. Nevertheless, such an approach was not possible within the time frame of this study.

A higher entropy meant that the Variant contained a more mixed set of Parts, which was interpreted as higher complexity. The entropy of this distribution was calculated according to Equation (2.9), but with the modification of normalizing the entropy such that it became dimensionless, bounded and comparable across Variants. The resulting equation then became:

$$H(X) = -\frac{\sum_{i=1}^{n} p(x_i)\log_2(p(x_i))}{\log_2(n)} \tag{3.1}$$

where n is the number of Part types and x_i denotes a particular Part type. This mathematical definition ensured that the Variant complexity was within $(0, 1]$. However, Variants with only one Part type would have undefined complexity according to Equation (3.1). Despite TTI's Variants not being of this nature, this phenomenon was undesired because it would mean that the complexity of entire platforms could be undefined. Following the interpretation of Variant complexity used in this thesis, a Variant with one Part type should be less complex than any other Variant. To model this, a minimum value of α was introduced into the formula. This allowed for equal comparisons between all sorts of platforms, and thus extended the scope of the metric. The minimum value was based on the characteristics of the platform under investigation and could be interpreted as the α-value of the least complex Variant possible for the given platform, with a margin factor of 250. The formula used to calculate this minimum α-value was defined as:

$$\alpha_{min} = -[\frac{250 \times \Omega - 1}{250 \times \Omega} \times \log_2(\frac{250 \times \Omega - 1}{250 \times \Omega}) + \frac{1}{250 \times \Omega} \times \log_2(\frac{1}{250 \times \Omega})] \tag{3.2}$$

where Ω denotes the maximum recorded number of Parts for any Variant in the platform. In this study Ω was on the order of thousands.

The margin factor of 250 was chosen so that even Variants with optimal distribution and $250 \times \Omega$ number of Parts, i.e. $250 \times \Omega - 1$ Parts in Part type A and one Part in Part type B, should still be considered for comparison. The particular value 250 was chosen so that, for this study, $250 \times \Omega$ was larger than one million total Parts, a quantity that is improbable in the context of trucks and their Variants. Thereby, all Variants in the platforms contributed fairly.

3.1.5 Structural complexity metric

Section 3.1.4 describes the assessment of Variant complexity that is used throughout this thesis. Combining that Variant complexity formula with the overall SC formula in Equation (2.3), the resulting expression becomes:

$$SC = C_1 + C_2C_3 = \sum_{i=1}^{N} \alpha_i + \left[\sum_{i=1}^{N} \sum_{j=1}^{N} \beta_{ij} A_{ij} \right] \gamma E(A) \tag{3.3}$$

where

$$\alpha_i = \begin{cases} -\frac{\sum_{k=1}^{n} p(x_k) \log_2(p(x_k))}{\log_2(n)}, & \text{if } \alpha > \alpha_{min} \\ \alpha_{min}, & \text{otherwise} \end{cases} \tag{3.4}$$

and

$$\beta_{ij} = f_{ij} \alpha_i \alpha_j \tag{3.5}$$

The SC formula in Equation (3.3) was applied to the context of TTI's platforms. Thus, f_{ij} in Equation (3.5) was defined as interface complexity, see Table 3.1. The interface complexity value was defined as $f_{ij} \geq 0$ where $f_{ij} \in \mathbb{R}$. The global scaling factor γ was defined as $\gamma > 0$ where $\gamma \in \mathbb{R}$.

3.2 Metric validation

The validity of the metric in Equation (3.3) was evaluated using Weyuker's properties, see Section 2.12. This was done using logical reasoning of each property. To test how the metric responded to identified driving factors from the literature and interview studies, it was applied to simulated platforms.

The metric's validity was further tested by comparing it with a simpler metric based on the different driving factors, specifically the total number of Variants, Parts, and SICs in the platform. This comparison was only performed on the simulated platforms. The simpler metric is shown in Equation (3.6).

$$C_{simple} = \sqrt[3]{N_{Variants} \times N_{Parts} \times N_{SICs}} \tag{3.6}$$

where $N_{Variants}$, N_{Parts}, and N_{SICs} denote the total number of Variants, Parts, and SICs within the platform, respectively.

The simulated platforms were modelled after the characteristics of the real platforms. This was beneficial because it allowed for quick stress testing over a larger sample space than what was available in terms of real platforms. Random platform configurations were constructed by randomly sampling platform parameters such as number of Variants, number of SICs, and KML length. These samples were drawn from the numerical range of each parameter. The ranges were chosen to reflect approximately the same proportions as TTI's platforms. This gave semi-accurate platform simulations as it did not account for the allocation of functionalities to each platform, i.e. two simulated platforms could have a large difference in total size.

However, the simulations were not intended to perfectly model TTI's platforms. The time-frame of constructing a more accurate model would exceed the time-frame of this thesis. Instead, this semi-accurate model was constructed and used for the purpose of generating different configurations that could test the metric for a multitude of special cases. It also allowed for quicker tuning of γ. However, final tuning had to be done for the real platforms because of the semi-accurate modelling of the simulated platforms. Due to confidentiality, the exact ranges of the platform parameters are not disclosed.

3.3 Case study

In order to develop and refine a method for applying the metric to a modular platform, an initial case study was conducted on a fraction of the ICE platform. This sub-platform consisted of the engine and exhaust modules, which were selected as a suitable starting point for method development and evaluation. The benefit of this approach was that the method could be developed and validated within a manageable scope before scaling. As long as the method proved to be scalable, the size of the sub-platform was not critical. A smaller case study required less data to process, which made it a practical initial step.

3.4 Full-scale platform

Once the method had been developed and refined through the case study, the next step was to apply it to a full-scale platform. The selected platform included two ICE Vehicle-PCs, and the interfaces were chosen from the predefined list of A-interfaces provided by Volvo, see Section 2.6.2. As TTI's platforms are very large, the main challenge in this step was obtaining accurate data with minimal manual effort, enabling the platform's complexity to be assessed as efficiently as possible. Sections 3.1.3.1 and 3.1.3.3 describe how the data was collected. Although parts of the data retrieval process were automated, see Section 3.4.1, some manual work was still required (e.g., all GICs were downloaded manually, as no automation script was developed for this purpose). Final tuning of γ was also executed in this process and the selected value was 0.1.

3.4.1 Automated data fetching

To streamline the data collection process described in Section 3.1.3.1, an automation script was developed for retrieving Variant-specific data from EDB. More specifically, the script automated the submission of Variants through the web form and the subsequent download of the generated CSV files containing the Parts associated with each Variant. Additionally, the script standardized the naming of downloaded files according to a consistent naming convention, reducing the amount of manual preprocessing required prior to the Shannon entropy calculations of each Variant.

3.4.2 Combining PCs

As explained in Section 2.5, some Variant Families are assigned their own PC, called Variant-PC. The Variants of the Variant-PCs that are included in a Vehicle-PC are documented in both PCs, meaning there is an overlap in listed Parts. However, not all Parts are included in this overlap. Vehicle-PC Variants mostly contain Parts related to the Parts that make up the Variant. Whereas Variants in Variant-PCs mostly consists of Parts related to installing the Variant in different vehicle types. To capture the full complexity of the platform, the Parts of shared Variants in Variant-PCs were incorporated in the Vehicle-PCs. More precisely, the pairwise intersection of Variants between the Vehicle-PCs and the engine, gearbox, front & rear axle, and cab Variant-PCs were extended with the Parts from these Variant-PCs. When extending these Variants, the overlapped Parts were ignored to exclude duplicates. This process is explained graphically in Figure 3.2.

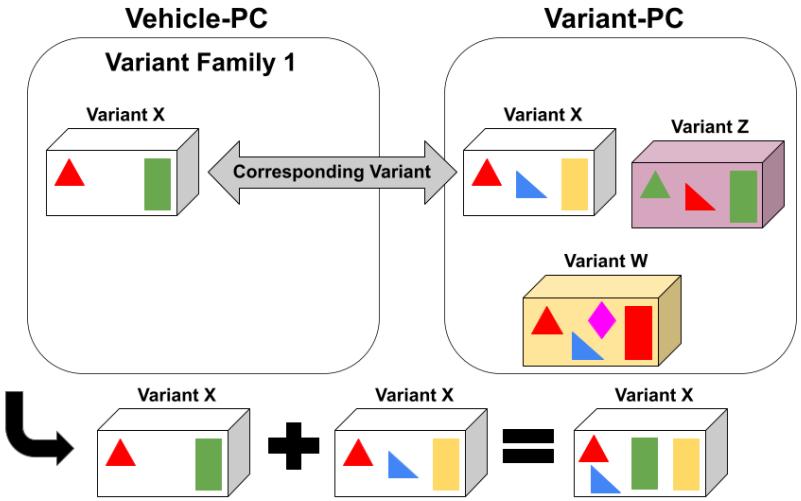

Figure 3.2: Figure explaining the process of combining Vehicle-PCs with Variant-PCs. The colored shapes represent different Parts. Matching Variants are merged together so that all Parts belonging to a Variant are accounted for. Only Variants that match in both PCs are merged. Overlapping Parts are ignored.

3.5 Comparing Platforms

Once the proposed method had been successfully applied to the full-scale ICE platform, see Section 3.4, an appropriate next step was to compare the ICE platform with the BEV platform. The purpose of this comparison was to evaluate how the metric could be used to identify and quantify differences in complexity between two platforms. To enable a fair comparison, the same methodology, parameters, and evaluation criteria were applied to both platforms. This was necessary because the complexity value of a platform alone does not provide meaningful insight. The value of the metric emerges when it is used comparatively between platforms.

4

Results

Chapter 4 encapsulates the results of the methods described in chapter 3. Due to the incremental nature of the methods used in this study, the results are made out of many subresults, where some are more extensive than others.

4.1 Findings from interview study

The interview data revealed several recurring themes in how platform complexity is understood and experienced in practice. Practitioners expressed their perspectives in different ways, understandably so, since members of the team work in different areas. Still, a consistent view emerged around a set of core drivers and characteristics of platform complexity.

The identified themes include: (i) interdependencies and system interactions, (ii) variability and product diversity, (iii) lack of standardization and unstable interfaces, (iv) architectural structure and modularization challenges, (v) organizational and process-related factors, and (vi) the dynamic and evolving nature of complexity. Each theme is described subsequently.

4.1.1 Platform complexity as interdependencies and system interactions

A central theme across responses is that platform complexity is largely driven by the number and nature of dependencies between modules and systems. Participants emphasized that complexity arises when changes in one segment of the platform trigger the need to evaluate many other interconnected components. In this view, complexity is not only a function of the number of elements, but also how tightly coupled they are and how interactions must be managed.

4.1.2 Variability and product diversity as key drivers

Another prominent theme is the role of variability in driving complexity. The platform must support a wide range of product variants and customer adaptations, which leads to a large number of possible configurations. While such variability is necessary to meet the diverse customer needs, it also introduces combinatorial challenges and increases the difficulty in maintaining a straightforward platform structure.

4.1.3 Lack of standardization and unstable interfaces

Several participants highlighted insufficient standardization as a major contributor to complexity. In particular, non-standardized or unstable interfaces between modules were described as problematic, as they hinder modularity and make changes more difficult to implement. The presence of such interfaces was perceived to increase both technical complexity and coordination effort.

4.1.4 Architectural structure and modularization challenges

Platform complexity was also linked to how the architecture is structured, including the definition of modules and their relationships. Participants noted that poor modularization, excessive one-to-one relationships between Variants, and unclear architectural boundaries contribute to increased complexity. In contrast, a well-structured architecture with clear module responsibilities was seen as a way to mitigate complexity.

4.1.5 Organizational and process-related factors

Beyond purely technical aspects, participants pointed to organizational factors as important drivers of complexity. Differences in ways of working across departments, lack of standardized processes, and insufficient documentation of decisions were all identified as contributing factors. In particular, decisions made at the project level rather than the platform level were perceived to increase long-term complexity.

4.1.6 Dynamic and evolving nature of complexity

Finally, platform complexity was described as dynamic rather than static. The continuous introduction of new technologies, systems, and requirements leads to an evolving set of dependencies and challenges. This makes complexity difficult to fully capture at a single point in time and highlights the need for approaches that can accommodate ongoing change.

4.2 The proposed metric

The metric that is proposed is adapted from Sinha and de Weck [31] and utilizes normalized Shannon entropy for assessing the Variant complexity α. The complete formula is presented in Equation (4.1).

$$SC = C_1 + C_2C_3 = \sum_{i=1}^{N} \alpha_i + \left[\sum_{i=1}^{N} \sum_{j=1}^{N} \beta_{ij} A_{ij} \right] \gamma E(A) \tag{4.1}$$

where

$$\alpha_i = \begin{cases} -\frac{\sum_{k=1}^{n} p(x_k) \log_2(p(x_k))}{\log_2(n)}, & \text{if } \alpha > \alpha_{min} \\ \alpha_{min}, & \text{otherwise} \end{cases} \tag{4.2}$$

and

$$\beta_{ij} = f_{ij} \alpha_i \alpha_j \tag{4.3}$$

4.3 Results from platform analysis

The preliminary platform analysis described in Section 3.1.2 illuminated the details of the platforms, such as the relationships between Variants and interfaces. One specific realization of this analysis was that each GIC connects only a subset of all Variant Families. The number of SICs per GIC is thus the total number of combinations of compatible Variants in the connected Variant Families. The large number of combinations per GIC revealed some of the platforms' structural complexity. The platform analysis also clarified how TTI's data was structured. These insights contributed significantly to the adaption of Sinha and de Weck's SC metric and Shannon entropy.

4.4 Simulated platform analysis

The tests using simulated platforms provided insight into how the metric behaved in a multitude of special cases. Figure 4.1 shows a selection of the simulation results. The proposed metric, Equation (4.1), showed a higher sensitivity compared to the simpler metric, Equation (3.6). This sensitivity is rooted in how the proposed metric evaluates interactions. Based on the proposed formula in Equation (4.1), platforms with large quantities of SICs and Variants, such as platforms 3, 4, and 6 should have higher values for the proposed metric. However, the bar chart in Figure 4.1 shows a large difference in the value of the proposed metric for these three platforms. At the same time, the simple metric values show little difference between the three platforms. This indicates that the proposed metric captures nuanced differences in platform architecture as opposed to the simple metric. The structural complexity of the platform arises from connectivity and interdependency which is hard to analyse from a macroscopic perspective. A comprehensive complexity measure can only be obtained by analysing all the connections in the platform at the element level, something that the proposed metric does by design. Because of this, the metric can

distinguish a low-complexity platform from a high-complexity platform despite no apparent difference from a macroscopic perspective. Furthermore, the number of Parts is a driving factor in the simple metric while the proposed metric is unaffected by this quantity. As shown in Figure 4.1, the number of Parts is large compared to the other values. By simultaneously observing the values of the simple metric, it can be noted that the simple metric does not fluctuate much between the different platforms; only a difference proportional to the total size of each platform exists. This suggests that the number of Parts masks the true complexity of the platform due to its large numerical value. A similar argument could be made for the number of SICs. However, since the interview and literature studies in Sections 3.1.1 and 2.11 identified connectivity and interdependency as key drivers of structural complexity, and since interfaces encapsulate these concepts while Parts do not, the number of SICs carries a higher complexity contribution than the number of Parts. This further suggests that the proposed metric is capable of prioritizing the main drivers of structural complexity without being perturbed by numerical noise in the platform. The simulation used the interface complexities in Table 3.1 and a γ-value of 0.01. More of the simulation results are shown in Appendix B.

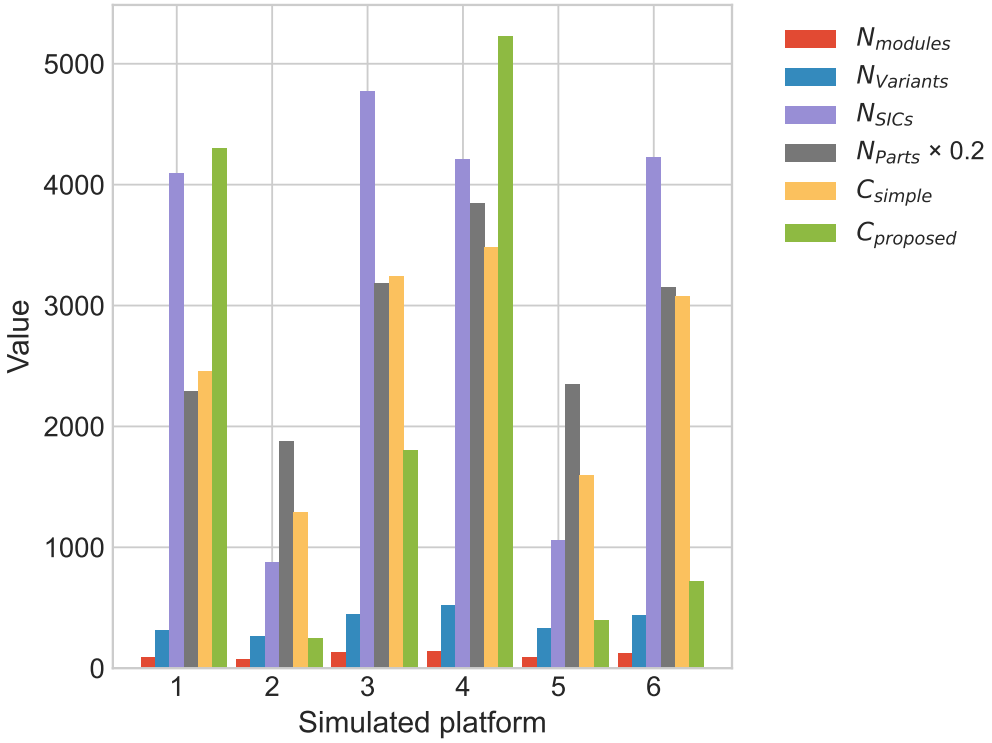

Figure 4.1: Bar chart showing the number of modules, Variants, SICs, and Parts alongside the proposed metric $C_{proposed}$ and simpler metric C_{simple} for six simulated platforms with randomly sampled parameter values. The number of Parts is scaled to 20% of its original value for readability.

4.5 Results from Weyuker validation

The results of the metric validation using Weyuker's properties [33] are presented in this section. Weyuker's properties, see Section 2.12, were applied to evaluate whether the proposed metric behaves as a meaningful measure of platform complexity. Each property was tested using logical reasoning.

Assume that we have two platforms, A and B. Let $SC(A)$ denote the scalar complexity value produced by applying the proposed metric to platform A.

For simplicity, let platform A consist of Variants V_1 and V_2, while platform B consists of Variants V_3 and V_4. Let α_i denote the complexity of Variant V_i, and let f_{ij} denote the interface complexity between Variants V_i and V_j.

- **Property 1:** There exist vehicle platforms A and B whose structural complexity differs, i.e. $SC(A) \neq SC(B)$.

  Proof: Assume that all Variant and interface complexities are equal between platforms A and B, except for one Variant such that $\alpha_1 \neq \alpha_3$. Since the metric contains the sum of the individual variant complexities, it follows directly that

  $$SC(A) \neq SC(B)$$

  $\implies$ The metric satisfies Property 1.

- **Property 2:** For any given complexity value c, only a finite number of vehicle platforms can have that value.

  Proof: Let the target structural complexity value be fixed to some constant c. Assuming that Variant complexities α_i and interface weights f_{ij} are represented with finite precision, only a finite number of combinations of Variants and interfaces can produce the value $SC = c$. Hence, there exist only finitely many platforms with the same structural complexity value.
  $\implies$ The metric satisfies Property 2.

- **Property 3:** Distinct platforms A and B may have the same structural complexity, i.e. $SC(A) = SC(B)$.

  Proof: Let

  $$\alpha_1 = \alpha_3, \qquad \alpha_2 = 2\alpha_4, \qquad f_{12} = \frac{f_{34}}{2}$$

  Then, although platforms A and B differ structurally, the resulting interaction contribution may satisfy

  $$SC(A) = SC(B)$$

  Thus, distinct platforms may produce identical structural complexity values.
  $\implies$ The metric satisfies Property 3.

- **Property 4:** Two platforms that provide the same functional capabilities may still differ in structural complexity, i.e. $SC(A) \neq SC(B)$.

  Proof: Two platforms may provide identical functionality while differing in structural arrangement or interface connectivity. Since the proposed metric

depends on both Variant complexity and interface interactions, different architectural structures may produce different complexity values despite equivalent functionality.
$\implies$ The metric satisfies Property 4.

- **Property 5:** For any two sub-platforms A and B, the complexity of the combined platform is greater than or equal to the complexity of each individual platform, i.e. $SC(A) \leq SC(A \cup B)$ and $SC(B) \leq SC(A \cup B)$.

  Proof: Let the combined platform of two sub platforms A and B be denoted $A \cup B$, with complexity $SC(A \cup B)$. Since neither Variant nor interface complexity can be negative and $\gamma > 0$, it must hold that combining subplatform A and B cannot result in a platform with less complexity than either subplatform individually. Hence, it follows that

  $$SC(A) \leq SC(A \cup B) \quad \text{and} \quad SC(B) \leq SC(A \cup B)$$

  $\implies$ The metric satisfies Property 5.

- **Property 6:** There exist platforms A, B, and O such that $SC(A) = SC(B)$, but when combined with platform O, the resulting complexities differ, i.e. $SC(A \cup O) \neq SC(B \cup O)$.

  Proof: Assume that

  $$SC(A) = SC(B)$$

  but that platform C introduces stronger interface interactions with A than with B. Since the interaction contribution depends on both adjacency and interface weights, it follows that

  $$SC(A \cup C) \neq SC(B \cup C)$$

  $\implies$ The metric satisfies Property 6.

- **Property 7:** Reordering modules within a platform may change its structural complexity. That is, there exist platforms A and B, where B is a permutation of A, such that $SC(A) \neq SC(B)$.

  Proof: Reordering Variants within a platform may alter the interaction structure represented by the adjacency matrix A_{ij}. Since the metric depends on these interactions, different arrangements of the same Variants may produce different structural complexity values.
  $\implies$ The metric satisfies Property 7.

- **Property 8:** Renaming or relabelling modules does not affect the structural complexity, i.e. if A is a renaming of B, then $SC(A) = SC(B)$.

  Proof: Renaming Variants without changing their complexity values or interface structure does not affect the proposed metric. Hence, the metric is invariant under renaming.
  $\implies$ The metric satisfies Property 8.

- **Property 9:** The complexity of a combined sub-platform may exceed the sum of the individual complexities, i.e. there exist A and B such that $SC(A) + SC(B) < SC(A \cup B)$.

Proof: Combining two platforms may introduce additional interface interactions between Variants belonging to the different platforms. These additional interactions increase the interaction contribution of the metric, such that

$$SC(A \cup B) > SC(A) + SC(B)$$

for certain platform combinations.
$\implies$ The metric satisfies Property 9.

In summary, all nine properties are fulfilled.

4.6 Results from case study

The case study, which focused on the engine and exhaust modules as described in Section 3.3, yielded a complexity value of 357.0893 using the proposed metric to combine interface count, component interdependencies, and configuration variability. The separate C_1, C_2, and C_3 values along with the total structural complexity are listed in Table 4.1.

Metric component	C_1	C_2	C_3	SC
Complexity value	29.348	64.263	5.1	357.0893

Table 4.1: Table showing the structural complexity values of the sub-platform investigated in the case study. The values are rounded to 4 significant figures.

4.7 Results from platform comparison

The resulting complexity value for the full-scale ICE platform is 279828.0788, representing a significant increase compared to the sub-platform used in the case study, which had a complexity value of 357.0893, presented in Section 4.6. This increase is disproportionately large which indicates that the proposed metric is sensitive to architectural aspects, such as connectivity and interdependency. To provide a reference point for the ICE platform's complexity, the BEV platform was also evaluated. The complexity value of the BEV platform is 12824.9153. All complexity values associated with each platform are listed in Table 4.2.

Platform	C_1	C_2	C_3	SC
ICE	268.4379	1504.6267	185.8	279828.0788
BEV	102.1619	287.8451	44.2	12824.9153

Table 4.2: Table showing the structural complexity values of the ICE and BEV platforms. The values are rounded to 4 significant figures.

4.8 Results from automated data fetching

The automation script for retrieving the Part data for each Variant within a given platform, described in Section 3.4.1, proved to significantly reduce the time required

for data collection. By comparing the time needed to manually download and rename a set of Variant files with the automated process for the same set of Variants, the automated approach was found to be approximately six times faster. The script also ensured that all downloaded files followed a consistent naming convention, reducing the amount of manual preprocessing required before performing the Shannon entropy calculations.

5

Discussion

This chapter will provide answers to respective research questions and discuss the findings. Reflections, limitations, and future work will also be discussed.

5.1 Answer to research questions

This section discusses the findings of the thesis in relation to the research questions presented in Section 1.3. Each research question is addressed individually based on the obtained results and the identified limitations of the study.

5.1.1 RQ1

How can module and interface variation be systematically measured to quantify architectural complexity in a platform?

The results suggest that module and interface variation can be used to systematically capture the structural complexity of a platform, where complexity is derived from the Variants' structure, connectivity between modules, and thus interface variability. The proposed metric captures how increases in module and interface variation introduce additional dependencies and interaction pathways, which are reflected in the C_2C_3 term, see Equation 4.1. This indicates that the metric captures not only the presence of system elements, but also the combinatorial growth of their interconnection, which is one of the primary drivers of architectural complexity. It differentiates between simple and complex architectures by assigning higher values to systems with denser and more interconnected structures. This aligns with the widely accepted view that complexity increases with the level of interdependence between system elements [31], as such systems are more difficult to analyse, modify, and scale. The findings therefore indicate that module and interface variation can be systematically incorporated into a structural complexity measurement of modular vehicle platforms, and thereby fulfilling RQ1.

5.1.2 RQ2

How can Volvo TTI's data fetching be structured to enable efficient and reliable data-driven analyses? Different data-fetching approaches were evaluated, such as API calls to Volvo's database and manual data fetching through EDB. Methods for automating use of data-fetching tools were investigated and improvements to

efficiency were found. A systematic and efficient pipeline for automatically fetching Variant data was then developed. This automated process is approximately six times faster than the manual process, see Section 4.8. The pipeline also includes validation and processing of product data. The pipeline enables efficient and reliable data-driven analyses.

5.1.3 RQ3

What complexity metrics can provide insight into the structural characteristics of modular vehicle platforms?

The thesis aimed to develop a complexity metric as a way to analyse TTI's modular platform architecture. As previously noted in Section 2.11.1, complexity can be defined in various ways and can therefore be measured differently, depending on how it is defined. In this work, multiple complexity metrics were explored. Sinha and de Weck's SC metric, see Section 2.11.3, was selected as the framework for the proposed metric. The SC metric is an established method used to assess the structural complexity of a system. This framework is suitable for assessing the complexity of modular vehicle platforms, where Variants and interfaces form a connected system to which the metric is applicable.

The results in Sections 4.4 and 4.7 show that the proposed metric is sensitive to structural changes in the architecture, such as adding, removing, or modifying Variants, introducing new interfaces, or reconfiguring existing connections. These changes alter the dependency structure of the system, which is reflected in the resulting complexity value. This sensitivity is important, since it ensures that even small architectural modifications are captured in the resulting complexity value.

While the metric produces a scalar value that may lack interpretability in isolation, it becomes meaningful in a comparative context. By evaluating multiple architectural alternatives, the metric enables relative assessment of complexity, which supports design decision-making. The lack of direct interpretability suggests that the metric is more suitable for comparison than for explaining the specific sources of complexity within a system. While this may seem to limit applicability outside TTI, the metric is data-driven and can be applied in any context, provided the same γ-value and a consistent convention for setting interface complexity are used. The important distinction is that the subjects to be compared must be of the same type, i.e. they have to be modular and possess the same number of module layers, see Section 2.2. For products with more than one module layer, the proposed metric has to be adapted to calculate each module layer recursively.

To assess the Variant complexity in TTI's platforms, entropy based methods were explored. Shannon entropy was chosen to represent Variant complexity because it gives a measure of complexity without having exact knowledge of each Variant's inherent physical structure. The SC metric was then augmented to utilize normalized Shannon entropy for the α-values. With the addition of a minimum α-value, as described in Section 3.1.4, the proposed metric satisfies the objective of RQ3.

5.2 Findings

This section presents the findings from the interview study, case study, and full-scale platform analysis, together providing a comprehensive understanding of structural complexity in the ICE and BEV platforms. The results show that the proposed structural complexity metric responds consistently to variations in system structure, particularly in terms of connectivity and interdependencies between modules.

5.2.1 Interview study

The findings from the interview study, case study and platform analysis together provide a deeper understanding of the platforms' characteristics. From the conducted interviews, a key observation is that the participants' perspectives aligned closely with the aspects captured by the proposed metric. Several of the themes identified during the interviews, such as interdependencies between modules, interface variability and architectural structure are directly related to the structural characteristics represented within the metric. The interview study highlighted that complexity is experienced not only as the number of modules or Variants, but through the relationships and dependencies between them. This supports the use of a structural complexity perspective, in which complexity is primarily associated with the connectivity and coupling between system elements [31]. The recurring emphasis on unstable interfaces and increasing configuration diversity further indicates that interface variation and architectural dependencies are important contributors to perceived platform complexity.

The provided understanding, however, remains limited to structural aspects of complexity. Organizational factors, evolving development processes, and temporal changes in the platform architecture were identified during the interviews but are not fully represented within the current metric formulation. The impact of these factors on overall complexity is discussed in Denis et al. [34] and Nagappan et al. [35]. Nagappan et al. connect organizational complexity directly to software quality, while Denis et al. highlight its effects on decision-making and knowledge flow in high-tech systems engineering. A further limitation of the interview study is the relatively small number of participating practitioners, which may restrict the breadth of perspectives captured. Although the interviews provided recurring and consistent themes, a larger sample of participants across additional roles and disciplines could strengthen the robustness of the findings.

5.2.2 Platform comparison

As shown in Table 4.2, the ICE platform exhibits higher complexity than the BEV platform, especially reflected in its larger C_2 and C_3 values. The ICE platform's C_2 value confirms a higher level of interaction complexity than the BEV platform. This could be because of the ICE platform having more complex interfaces on average, or it could be caused by complex Variants interacting more often on average, see Equation (4.3). Another reason for this could be that the ICE platform has a higher number of total connections compared to the BEV platform. As shown in

Equation (4.1), C_2 does scale with number of connections, though not as directly as C_3. By comparing the C_3 values of the two platforms in Table 4.2, it can be concluded that the latter reason is the most probable because the ICE platform has more than four times the connectivity of the BEV platform. This reveals a higher degree of interdependence and connectivity between modules and Variants within the platform. Another argument for why the difference in C_2 is driven by connectivity is that the quotient between the C_2 values of the two platforms is roughly 5.2, and the quotient between the C_3 values is roughly 4.2. This indicates that the difference in C_2 value between the platforms is most likely driven by difference in connectivity, rather than difference in interface complexity. This is further strengthened by the fact that both platforms are relatively similar in architecture, i.e. they are both truck platforms and share many Variants. Thus, the average difference in interface complexity between the platforms is not significant enough to suggest that the interface complexity is the driving factor in the elevated C_2 value of the ICE platform. For the interface complexity to be the driving factor, it would require the ICE platform to have a majority of its interfaces at a complexity value of four, which is reserved for the Master Reference GIC class, see Table 3.1. Furthermore, it would require that the contribution of connectivity in C_2, see Equation (4.1) is being negated by connections between low complexity Variants, see Equation (4.3).

This high connectivity in the ICE platform likely stems from it having modules with more performance steps than the BEV platform, a difference probably caused by its longer development history and historical role in Volvo's profits. Over time, the introduction of additional configurations and refinements has likely increased the number of module interfaces and dependencies, contributing to greater architectural complexity compared to the more recently developed BEV platform. However, it is important to note that the proposed metric is not meant to serve as a direct comparison between ICE and BEV platforms, as these are not competing alternatives at Volvo but rather coexisting architectures. Instead, the metric is best utilized when either platform is being further developed, enabling evaluation of new alternative designs from a structural complexity perspective.

The case study and the full-scale platform analysis further strengthened the understanding gained in the interview study by demonstrating how these characteristics can be represented quantitatively through the proposed metric. The observed increase in complexity from the sub-platform case study to the full platform analysis, mentioned in Section 4.7, suggests that the metric is capable of reflecting the scaling effects of increasing interdependencies, interfaces and configuration diversity across the platform.

5.2.3 Complexity metric

The SC metric has, however, limitations that should be acknowledged. Although the proposed metric captures structural properties such as connectivity, coupling, and interface interdependencies, it does not fully represent all dimensions of complexity present in a modular platform architecture. One important limitation is that the metric is fundamentally static and structural in nature. It evaluates the architecture based on the existence and weighting of interfaces between modules, but it does not

capture dynamic behaviours such as system evolution over time, operational interactions, manufacturing variability or lifecycle effects. As a result, two architectures with similar structural characteristics may exhibit significantly different performance, robustness or development complexity in practice.

Another limitation concerns the subjectivity involved in assigning complexity values to interfaces. The metric depends heavily on how interfaces are defined, categorized, and weighted. Since the interface variation data provided by TTI formed the basis of the analysis, the resulting complexity values are sensitive to modelling assumptions and engineering judgment. Different decompositions of the platform, or different interface classifications, could therefore produce different complexity outcomes. This limits the objectivity and repeatability of the method. To ensure valid comparisons, all platforms must be evaluated with the same parameter values, such as γ.

Despite these limitations, the proposed metric provided a useful quantitative basis for comparing platform alternatives and identifying drivers of structural complexity within TTI's modular platforms.

Future work could address these limitations in several ways. One direction would be to complement the proposed metric with additional dimensions of complexity, such as functional, organizational, manufacturing or lifecycle complexity. Incorporating dynamic aspects of the platform, such as temporal dependency modelling, could provide a more comprehensive representation of system behaviour. Future studies could also improve the objectivity of the metric by developing standardized methods for interface weighting. Moreover, as mentioned in Section 3.1.4, further improvements to precision could be made by investigating different ways to assess the complexity of a platform's Variants, one of which is to rigorously analyse each Variant using an integrated architecture metric as in Raja et al. [30].

5.2.4 Simulated platform analysis and Shannon entropy

The analysis of the simulated platforms validated the normalization of Shannon entropy, as the large number of Parts combined with non-normalized Variant complexity could render the metric unstable and undermine its comparative use across different platforms. Without normalization, the metric values could grow excessively, making cross-platform comparison meaningless. Furthermore, the comparison between the proposed metric and the simpler metric on the simulated platforms demonstrated that the proposed metric exhibits greater sensitivity to changes. A γ-value of 0.01 was selected for the simulated platforms to balance out the contributions of C_1 and C_2C_3. Later when assessing the full-scale platform, this value was found to be too small for the real platforms as they had lower connectivity and utilized more standardization than the simulated platforms, requiring a higher γ of 0.1.

While Shannon entropy is not well established in the field of hardware platform complexity, its theoretical properties make it well-suited for this application. Its use in deriving the α parameter in our proposed metric, Equation (4.1), provides a systematic, reproducible basis for complexity quantification across platform Variants. The findings of this thesis suggest that entropy-based complexity assessment is both

feasible and meaningful in this domain. By normalizing the formula, as shown in Equations (3.1) and (3.4), the implications on complexity changes to not scale with the number of Parts a Variant contains, but instead reflect how variation is distributed across them. This aligns with this thesis's interpretation of complexity: a system is not inherently more complex simply because it contains more element types. This ensures the metric captures the shape of variation rather than its absolute magnitude, enabling meaningful comparison across Variants of different sizes. This augmentation is a contribution to research of modular platform complexity as it has not been adapted in this sense prior to this study. By extension, this view allows comparisons between all types of products because it focuses on the level of optimization of the product, which is beneficial from a universal measurement perspective. However, this view is not superior in all cases because by scaling Variant complexity with the number of Part types, complexity comparisons between similar products could be more precise.

As stated in Section 2.11.4, Variant complexity does not scale with the total number of Parts in a Variant. Nonetheless, the number of Parts does affect the granularity of the probability distribution of Part types. This means that two Variants with the same distribution and amount of Part types, but different amounts of total Parts can have slightly different Shannon entropies and thus Variant complexities. This has no negative effect on the proposed metric, it only means that Variants with fewer Parts are, from a complexity perspective, more sensitive to updates.

The normalization of the Shannon entropy also produced a limitation initially. As explained in Section 3.1.4, Variants with one Part type would be undefined. This has no effect for TTI's platforms, because their current Variants contain more than one Part type. Nevertheless, it is an undesirable effect, because it theoretically allows platforms to have undefined complexity in cases where each Variant consists of only one Part type. To handle these cases, the minimum α-value is assigned to Variants with only one Part type, see Equations (3.2) and (3.4). Even if this case is improbable in a real Variant, it must be considered for the sake of the metric's scope, as mentioned in Section 3.1.4.

5.2.5 Weyuker validation

The proposed metric satisfies Weyuker's properties, see Section 4.5, providing a foundational validation. However, fulfilling these properties alone does not guarantee usefulness [36]. Nevertheless, taken together with its sensitivity to structural changes and its alignment with the intended purpose, satisfying these properties indicates that the proposed metric is useful. Altogether, these observations suggest that the proposed metric is effective in capturing how module and interface variation contribute to architectural complexity. Its sensitivity to connectivity and configuration enables it to reflect key properties of platforms, even though it does not explicitly capture functional or dynamic aspects.

Taken together, the qualitative and quantitative findings suggest that structural complexity is a useful way to represent perceived platform complexity and that the

proposed metric captures this structural complexity.

5.3 Reflections

When developing a new modular product architecture, there are multiple design decisions, such as the addition of interfaces or modules, among others, to be made with differing advantages. The number of combinations of these decisions defines all possible platforms. With the proposed metric, a ranking of the structural complexity of each platform can be made. As complexity is commonly associated with cost, this assessment can then be one of the arguments for or against a particular platform design. Because complexity usually hinders scalability, the complexity argument might be more important for products that are large or have a long design life cycle, or both. Such an argument could be made without this metric but never with the same objectivity; without it, such assessments risk ambiguity and often require extensive experience and expertise. As complexity depends on interpretation, the proposed metric should not be used in isolation as a definitive measure of complexity. It should instead be used in combination with other complexity metrics and methods to make balanced and robust design decisions.

An example of modelling assumptions and subjectivity in this study is the ranking of GIC class complexities, as shown in Table 3.1. It was based on an importance hierarchy in internal TTI documentation [4]. This hierarchy focused on the level of change-propagation of the interface classes, rather than the inherent interface complexity. Despite this, it is a useful ranking because the platform analysis, explained in Section 3.1.2, found that almost all GIC classes were geometric and therefore difficult to distinguish from each other in terms of inherent complexity. Even if some GICs had other natures, their instances were few and would thus not significantly contribute to the total platform complexity. Because of this, and the fact that interdependency is a measure of structural complexity, the change-propagation hierarchy was used as a foundation for the GIC class ranking.

As mentioned in Section 3.1.3.2, the assigned GIC complexity values were primarily intended to reflect the relative differences in complexity between interfaces, particularly with respect to their level of interdependency. Due to the limited time frame of the study, integer values were used as a simplified representation of interface complexity. However, these values are still beneficial as they facilitate the analysis of which interfaces drive structural complexity. Future work could investigate more systematic and accurate approaches for determining interface complexity values.

In this study, only A-interfaces were considered, as stated in Section 2.6.2. This is because they are the most important from a platform perspective. By including B- and C-interfaces in the structural complexity calculation, higher precision could potentially be achieved. However, since the investigated platforms are in development, these less regulated interfaces could also introduce errors in the complexity assessment. In addition, this study is based on two platforms and thus a limited set of modules and interfaces, which restricts the generalizability of the findings.

Only physical aspects of modular vehicles are assessed in this thesis. However, in

modern vehicles such as Volvo's trucks, software is also a prominent part of the product. Because software development and deployment tend to be quicker and more wide-ranging than their hardware counterparts, complexity can be even more problematic in the software domain. A comprehensive analysis encapsulating both domains would be extremely useful for products with high interplay between hardware and software, such as modern trucks or airplanes. Nevertheless, the time frame and scope of this thesis were too limited to permit such an analysis.

Future studies of TTI's platforms could take B- and C-interfaces into account for a potential gain in accuracy, preferably for discontinued platforms as these interface categories are more unstable than A-interfaces, which can introduce errors for platforms in development. It would also be useful to analyse the interplay between soft- and hardware in modular vehicles. Future work could expand the interview study's participant base to include a wider range of stakeholders involved in platform development and maintenance. The metric could also be extended to better incorporate dynamic and organizational dimensions of complexity, as well as validate the approach using larger datasets and additional platforms.

Other insights can also be gained about the success of platforms. For example, by analysing the correlation between complexity and profit margins of products based on the given platform. This may then aid in the optimization of platform lifespan and scope.

6

Conclusion

This thesis investigated how module and interface variation can be used to quantify architectural complexity in modular vehicle platforms at Volvo TTI. By combining structural complexity theory with entropy based assessment of Variant complexity, the work aimed to develop a quantitative approach for assessing architectural complexity in modular vehicle platforms. The proposed metric enables platform analysis and thereby supports architectural decision-making.

The results demonstrated that the proposed metric responds consistently to structural changes in the platform architecture, particularly an increase in connectivity, interface variation and interdependencies between modules. The metric therefore captures important structural characteristics associated with architectural complexity. The integration of normalized Shannon entropy enabled Variant complexity to be quantified without requiring detailed physical modelling of each Variant. In addition, the developed data-fetching pipeline enabled efficient large-scale analysis of platform data.

The thesis contributes to a quantitative approach for assessing structural complexity in modular vehicle platforms by combining an established SC metric with entropy-based Variant assessment. The work also demonstrates how platform data can be systematically processed to support data-driven architectural analysis in an industrial context.

The proposed metric only quantifies structural complexity and therefore does not capture dynamic, organizational, or lifecycle-related aspects of complexity. Furthermore, the resulting complexity values are influenced by modelling assumptions and engineering judgment regarding interface definitions and weighting. In order to make balanced and robust design decisions, the proposed metric should be used in combination with other complexity metrics and methods.

Future work could extend the framework by incorporating dynamic and organizational dimensions of complexity, refining interface weighting methodologies, and validating the approach across additional modular platforms and industrial contexts. A more exact measure of Variant complexity could be achieved by investigating the physical structure of each Variant using established methods for integrated architecture. Furthermore, incorporating complexity assessment of product software could capture a broader view of which factors drive complexity in the product as a whole.

Despite its limitations, the proposed framework demonstrates that structural complexity in modular vehicle platforms can be systematically quantified using module

and interface variation. The approach provides a useful basis for comparative architectural assessment and can support future platform development by enabling more objective complexity-informed design decisions. The consistency between the metric's behaviour and expected structural changes also indicates that it reflects perceived platform complexity reasonably well.

Bibliography

[1] Jaime Alberto Mesa, Iván Esparragoza, and Heriberto Maury. Modular architecture principles – maps: a key factor in the development of sustainable open architecture products. *International Journal of Sustainable Engineering*, 2019. doi: 10.1080/19397038.2019.1634157.

[2] Ernst Fricke and Armin P. Schulz. Design for changeability (DfC): Principles to enable changes in systems throughout their entire lifecycle. *Systems Engineering*, 8(4):279–341, 2005. ISSN 1098-1241. doi: 10.1002/sys.20039.

[3] Karl T. Ulrich. Fundamentals of product modularity. In Sriram Dasu and Charles Eastman, editors, *Management of Design*, pages 219–231. Springer, Dordrecht, 1994. doi: 10.1007/978-94-011-1390-8_12.

[4] Volvo TTI. Internal documentation, 2026. Internal company documentation and education, not publicly available.

[5] Kaushik Sinha and Olivier L. de Weck. Structural complexity quantification for engineered complex systems and implications on system architecture and design. In *Proceedings of the International Design Engineering Technical Conferences and Computers and Information in Engineering Conference (IDETC/CIE)*, 2013.

[6] Carliss Y. Baldwin and Kim B. Clark. *Design Rules, Vol. 1: The Power of Modularity*. MIT Press, Cambridge, MA, USA, 2000.

[7] Michael Erbschloe. Structured programming. *Salem Press Encyclopedia*, 2021. Available through EBSCOhost.

[8] Carliss Y. Baldwin and Kim B. Clark. *Design Rules, Volume 1: The Power of Modularity*. MIT Press, Cambridge, MA, 2000.

[9] David L. Parnas. On the criteria to be used in decomposing systems into modules. *Communications of the ACM*, 15(12):1053–1058, 1972. doi: 10.1145/361598.361623.

[10] Daniel J. Sturtevant. *System Design and the Cost of Architectural Complexity*. PhD thesis, Massachusetts Institute of Technology, 2013.

[11] Landívar Chávez and José Luis. Complexity cost analysis in a large product line. Master's thesis, Massachusetts Institute of Technology, 2006.

[12] Karl Ulrich. The role of product architecture in the manufacturing firm. *Research Policy. May, 1995, Vol. 24 Issue 3, p419, 30 p.*, 1995.

[13] Marc-Antoine Roy and Georges Abdul-Nour. Integrating modular design concepts for enhanced efficiency in digital and sustainable manufacturing: A literature review. *Applied Sciences, 2024, 14(11), 4539*, 2024. doi: 10.3390/app14114539.

[14] Tarek AlGeddawy. A dsm cladistics model for product family architecture design. *Procedia CIRP, 21, 2014*, 2014. doi: 10.1016/j.procir.2014.03.122.

[15] Stéphanie Bouchard, Sébastien Gamache, and Georges Abdulnour. Operationalizing mass customization in manufacturing smes—a systematic literature review. *Sustainability 2023, 15(4), 3028*, 2023. doi: 10.3390/su15043028.

[16] Jarkko Pakkanen, Tero Juuti, and Timo Lehtonen. Identifying and addressing challenges in the engineering design of modular systems – case studies in the manufacturing industry. *Journal of Engineering Design, vol. 30, 2019*, 2018. doi: 10.1080/09544828.2018.1552779.

[17] Mohamed Kashkoush and Hoda ElMaraghy. Designing modular product architecture for optimal overall product modularity. *Journal of Engineering Design, Vol.28 2017*, 2017. doi: 10.1080/09544828.2017.1307949.

[18] David Robertson and Karl Ulrich. Planning for product platforms. *Sloan Management Review. Summer98, Vol. 39 Issue 4, p19-31. 13p. 12 Black and White Photographs, 3 Diagrams, 4 Charts, 1 Graph.*, 1998.

[19] Yukihisa Otani, Yamauchi Takashi, Jaan Praks, and Mengu Cho. A dsm-based framework for design complexity and component criticality in cubesat missions. *CEAS Space Journal*, 2026. doi: 10.1007/s12567-026-00713-3.

[20] Steven D. Eppinger and Tyson R. Browning. *Design Structure Matrix Methods and Applications*. MIT Press, Cambridge, MA, 2012.

[21] Udo Lindemann. *A Vision to Overcome "Chaotic" Product Development Processes by Supporting Design Structuring with the Design Structure Matrix Method*. PhD thesis, Technical University of Munich, 2009.

[22] Hamdi A. Bashir and Vince Thomson. Estimating design complexity. *Journal of Engineering Design. Sep99, Vol. 10 Issue 3, p247-257.*, 1999. doi: 10.1080/095448299261317.

[23] Nam P. Suh. A theory of complexity, periodicity and the design axioms. *Research in Engineering Design (1999)11:116–131*, 1999. doi: 10.1007/PL00003883.

[24] Günther Schuh, Stefan Rudolf, and Till Vogels. Performance measurement of modular product platforms. *Procedia CIRP 2014 17:266-271*, 2014. doi: 10.1016/j.procir.2014.02.028.

[25] Herbert A. Simon. The architecture of complexity. *Proceedings of the American Philosophical Society, Vol. 106, No. 6. (Dec. 12, 1962), pp. 467-482.*, 1962.

[26] Olivier Serrat. *Understanding Complexity*, pages 345–353. Springer Singapore, Singapore, 2017. ISBN 978-981-10-0983-9. doi: 10.1007/978-981-10-0983-9_39. URL `https://doi.org/10.1007/978-981-10-0983-9_39`.

[27] Alfonso Lanza, Joshua Sutherland, Marc-Andre Chavy-Macdonald, and Olivier L. de Weck. Quantifying structural complexity, effort, and performance: An early experiment using network design tasks. *Systems Engineering*, 29(3):387–405, 2026. doi: 10.1002/sys.70027.

[28] Olivier L. de Weck, Daniel Roos, and Christopher L. Magee. *Engineering Systems: Meeting Human Needs in a Complex Technological World*. The MIT Press, Cambridge, Massachusetts, 2011. ISBN 978-0-262-01670-4.

[29] Joshua D. Summers and Jami J. Shah. Mechanical engineering design complexity metrics: size, coupling, and solvability. *Journal of Mechanical Design. Feb, 2010, Vol. 132 Issue 2*, 2010. doi: 10.1115/1.4000759.

[30] Visakha Raja, Michael Kokkolaras, and Ola Isaksson. A simulation-assisted complexity metric for design optimization of integrated architecture aero-engine structures. *Structural and Multidisciplinary Optimization, Vol. 60 2019*, 2019. doi: 10.1007/s00158-019-02308-5.

[31] Kaushik Sinha and Olivier L. de Weck. A network-based structural complexity metric for engineered complex systems. *2013 IEEE International Systems Conference (SysCon)*, 2013. doi: 10.1109/SysCon.2013.6549917.

[32] Claude Elwood Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 1949. doi: 10.1002/j.1538-7305.1948.tb01338.x.

[33] Elaine J. Weyuker. Evaluating software complexity measures. *IEEE Transactions on Software Engineering*, 14(4):480–491, 1988. doi: 10.1109/32.4668.

[34] Sherly Agnes Reni Denis, Marcus Vinicius Pereira Pessoa, and G. Maarten Bonnema. Effects of organizational complexity on high-tech systems design decisions: Insights from a dutch equipment manufacturer. In *2025 20th Annual System of Systems Engineering Conference (SoSE)*. IEEE, 2025.

[35] Nachiappan Nagappan, Brendan Murphy, and Victor R. Basili. The influence of organizational structure on software quality. Technical Report MSR-TR-2008-11, Microsoft Research, Cambridge, UK, 2008.

[36] Matthias Kreimeyer and Udo Lindemann. *Complexity Metrics in Engineering Design: Managing the Structure of Design Processes*. Springer Berlin, Heidelberg, 2011. doi: 10.1007/978-3-642-20963-5.

A

Interview notes

A.1 Interview questions

The following questions were used as a basis for the interviews:

- What is platform complexity from your perspective?
- What are the driving factors that make a platform complex?
- Does complexity impact your work, and if so, how?

A.2 Interview responses

Person 1

Complexity arises when there are many unique installations. When you cannot standardise key interfaces, things become complex. Complexity appears when changing something requires considering a very large number of dependencies. You cannot change anything without affecting or redesigning everything.

Person 2

The industry is complex because we have many different trucks (BEV / ICE / fuel cell | Renault, low-cost and high-cost variants). There is a lot of variability.

Different departments also have different structures (steering, autonomous, brakes), and different ways of working. A product structure needs to be flexible to enable introduction of new technologies.

To reduce complexity, ways of working need to become more standardised.

Person 3

A truck is complex because there are interactions between many systems (steering, propulsion, suspension). There are many different types of systems (thermal, signal, braking).

Complexity comes from:

- Interactions with many systems
- Large number of parts and part types
- The type and amount of interactions between systems

- Life cycle aspects
- Legal and regulatory requirements

Combinations and variability are key drivers of complexity.

Person 4 Complexity is based on the number of possible variations. It is important to design things in a smart way and understand all constraints.

To reduce complexity, as few modules as possible should be used while still fulfilling customer requirements, and also allowing differentiation between brands (e.g. Volvo and Renault).

Person 5

Platform complexity relates to modules and interfaces. Complexity is the number of interfaces we follow based on our role and responsibility, and all modules connected to those interfaces.

Additional complexity arises from dependencies that appear outside the defined structure. These must be considered, for example fuel cell systems, which means the system is never static.

A key question is how much needs to be kept in mind when introducing something new (again, dependencies).

Many variants increase complexity: when introducing a new variant or making a change, questions arise such as "should we keep the existing solution?", which is not always visible directly in the interfaces.

On variant-driven complexity, a possible driver could be: If there are many unique 1:1 relationships (one module only works with one other module), or if only one architecture/product uses something, it should either be phased out or expanded.

Person 6

It is the gap between intended customer variation and unintended technical growth.

We want to support variation for customer needs, but this creates complex combinations in practice.

Variation is introduced at different levels. Stable interfaces are needed to separate these layers.

Documentation complexity also emerges from this. The introduction of software makes the system even more complex.

Weak or unstable interfaces, lack of structure, and lack of standardised solutions.

Documentation and knowledge structure are also issues - often it is unclear why certain decisions were made because they were not documented.

Project-level decisions instead of platform-level decisions increase complexity.

Customer adaptations (mods) often modify the product at a low level, meaning they are not always documented.

A.3 Closing remark

The interviews were conducted orally and summarized in written form during the sessions.

B

Simulation results

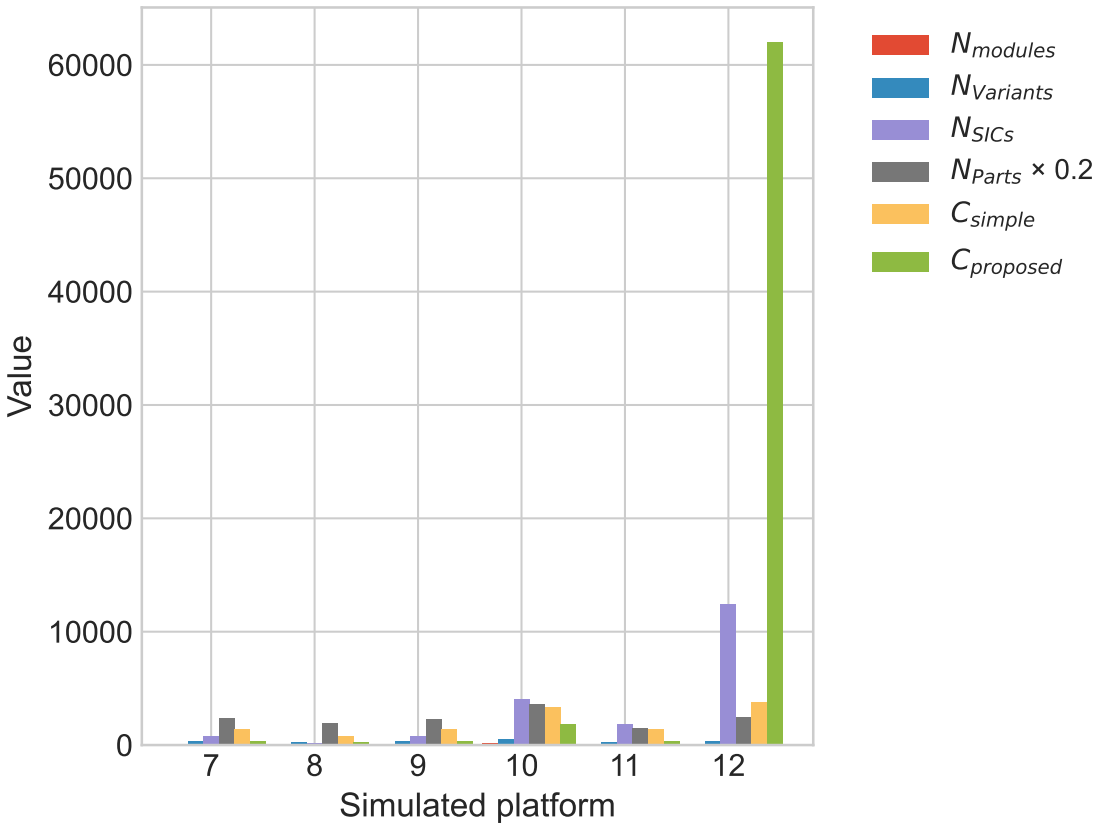

Figure B.1: Bar chart showing the number of modules, Variants, SICs, and Parts alongside the proposed metric $C_{proposed}$ and simpler metric C_{simple} for six simulated platforms with randomly sampled parameter values. The number of Parts is scaled to 20% of its original value for readability.

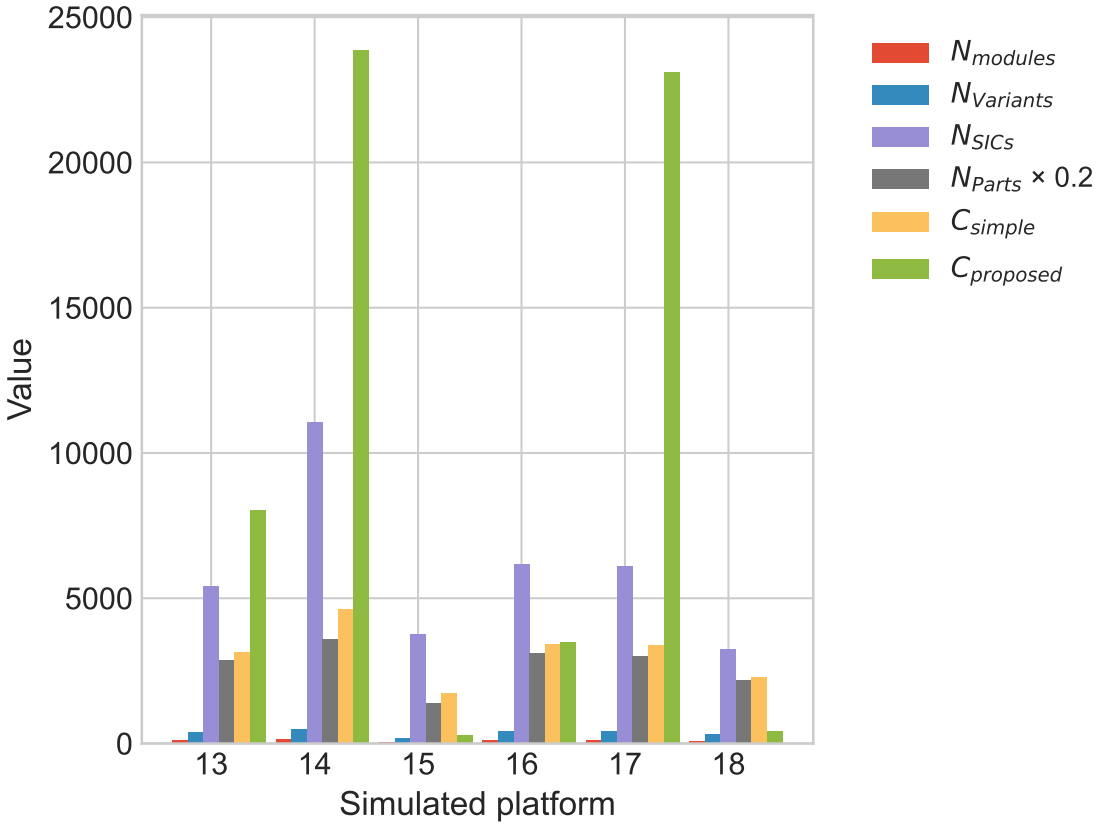

Figure B.2: Bar chart showing the number of modules, Variants, SICs, and Parts alongside the proposed metric $C_{proposed}$ and simpler metric C_{simple} for six simulated platforms with randomly sampled parameter values. The number of Parts is scaled to 20% of its original value for readability.

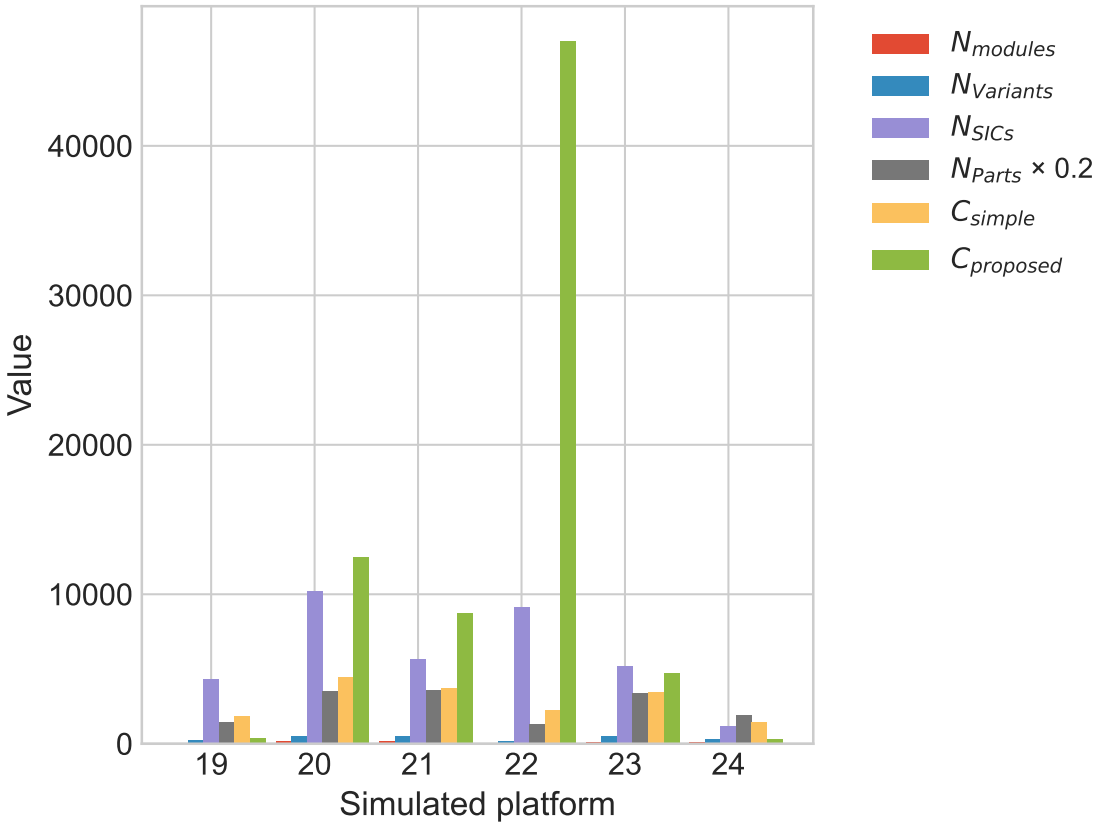

Figure B.3: Bar chart showing the number of modules, Variants, SICs, and Parts alongside the proposed metric $C_{proposed}$ and simpler metric C_{simple} for six simulated platforms with randomly sampled parameter values. The number of Parts is scaled to 20% of its original value for readability.

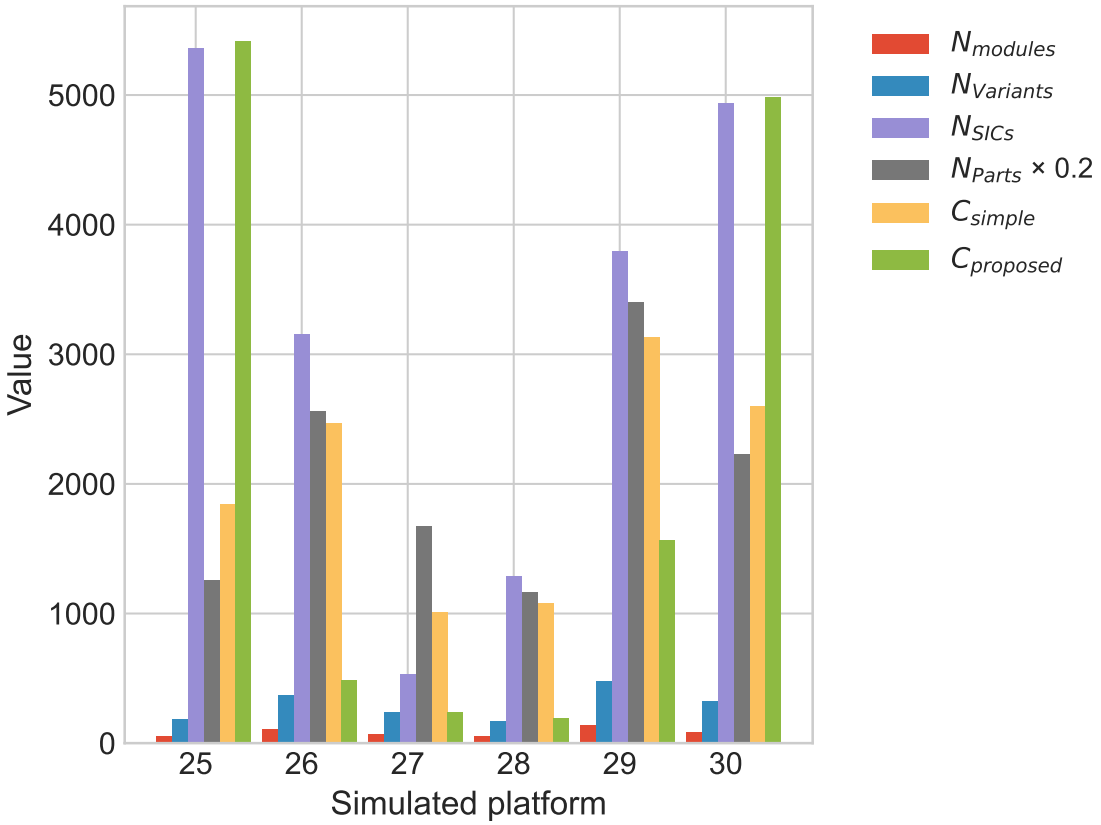

Figure B.4: Bar chart showing the number of modules, Variants, SICs, and Parts alongside the proposed metric $C_{proposed}$ and simpler metric C_{simple} for six simulated platforms with randomly sampled parameter values. The number of Parts is scaled to 20% of its original value for readability.

Department of Mechanics and Maritime Sciences
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY